The Importance of Using the Linear Transformation Matrix in Determining the Number of Processing Elements in 2-dimensional Systolic Array for the Algorithm of Matrix-Matrix Multiplication

Halil Snopce1, Lavdrim Elmazi2
1, 2 South East European University CST Faculty, 1200 Ilindenska bb, Tetovo, Macedonia
 1h.snopce@seeu.edu.mk  2l.elmazi@seeu.edu.mk

Abstract. Matrix multiplication plays very important role in many scientific disciplines because of fact that it is considered as the main tool for many other computations in different areas, like those in seismic analysis, different simulations (like galactic simulations), aerodynamic computations, signal and images processing etc. In this paper is using a special design named systolic arrays which are suitable for matrix multiplication algorithm and offer both pipeline ability and parallelism. On the area of systolic designs there are two main questions: the first one is how to choose the appropriate systolic array for certain application and the second question is how to minimize the number of processors. The main result in this work gives a possible answer for the second question mentioned above. In this work is given the so called transformation matrix, which maps the given index space in another index space. Transformation used in this new index space reduces the number of processing elements in the array. We illustrate all possible instances of transformation matrices and we show the importance of using the transformation matrix by comparing the number of processing elements of the array where we use it with another array where this transformation is not used. For this purpose is given a mathematical explanation using theorems 1 and 2. The comparison is made using the matrices of size N=4.

Keywords: Systolic array, matrix multiplication, linear transformation matrix, number of processor elements, index space, projection direction, position of processing elements (PE), communication links, data flow, 3-nested loop algorithm.

1. Introduction

A group of researchers, headed by Kung, has introduced a systolic concept of parallel computing [19-20]. One of the most challenging problems is development of methodology for mapping an algorithm into a systolic array. Moldovan and Fortes [24] and Miranker and Winkler [25] worked on how to minimize the computation time of systolic array. Kung [20] used matrix multiplication to demonstrate his concept of systolic arrays for computationally intensive problems. The problem of multiplying two matrices has been studied extensively and a number of arrays have been proposed for this purpose [9,20,26]. SHSA (standard hexagonal systolic array) is originally proposed by Kung. Evans has proposed array named FHSA (fast hexagonal systolic array)[17], where for N=4 the number of PEs (processor elements) is 23 (better than the array SHSA which we use in our analysis). Milovanovic [4,18] has proposed the array MHSA (Modified hexagonal systolic array) where for N=4 the number of PEs is 16 (The same number like array used in our analysis with using linear transformation matrix L).

2. Definition of systolic arrays

Definition 1: [given by Rao-[11] ]- A systolic array is a network of processors in which the processors can be placed at the grid points of a finite lattice so that:
-Topologically: If there is directed link from the processor at location \( I \) to the processor at location \( dI \) for some \( d \), then there is such a link for every \( I \) within the lattice.
-Computationally: If a processor receives a value on an input link at time \( t \), then it receives a value at time \( t + \Delta \) on the corresponding output link, where \( \Delta \) is time period that is independent of the network size, the orientation of the link and the location of the processor.

Definition 2: [given by Kung-[12]]-A systolic array is a computing network possessing with the following features:
-Synchrony: A systolic array is controlled and synchronized by a global clock with fixed length clock cycles. Data are rhythmically computed and passed through the network.

-Modularity and regularity: A systolic array consists of modular processing units with homogeneous interconnections and the computing network can be extended indefinitely.

-Spatial locality: A systolic array manifests a locally communicative interconnection structure (spatial locality). Each cell or processing element only communicates with its neighboring cells.

-Pipelinability: A systolic array exhibits a linear rate pipelinability to speed up processing rate.

-Repeatability: In most systolic arrays, the systolic network is usually the repetition of one single type of cell and interconnection. Even in the over cases, there are usually at most two or three different types of systolic cells involved in the network.

-Parallel processing: With systolic arrays can be achieved higher parallelism.

Taking on consideration that matrix multiplication satisfies characteristics of above definitions, systolic arrays could be used for this purpose (for short information about characteristics, properties, models and data flow of systolic arrays use also [14], [15] and [16]).

3. Systolic array for matrix multiplication using linear transformation matrix

Let A and B be two matrices of size N x N and we consider the problem of finding the resulting matrix C using the algorithm for matrix multiplication given below:

\begin{align*}
\text{Algorithm 1:} \\
\text{for } i, j, k = 1 \text{ to } N \\
da(i, j, k) &= a(i, j - 1, k) \\
b(i, j, k) &= b(i - 1, j, k) \\
c(i, j, k) &= c(i, j, k - 1) + a(i, j, k - 1) \cdot b(i, j, k - 1) \\
\text{end where } \\
da(i, 0, k) &= a_i, b(0, j, k) = b_{ij}, c(i, 0, 0) &= 0
\end{align*}

Let \( P_{ind} = \{(i, j, k) / 1 \leq i, j, k \leq N\} \) be index space of used and computed data for matrix multiplication. In fact it is a set of all lattice points enclosed within a specified region in 3-dimensional Euclidean space. So, in the given algorithm \( a(i, j, k) \) denotes a position of variable \( a \) in this defined space. (In the defined algorithm for matrix-matrix multiplication there are variables for matrixes A, B and resulting matrix C, denoted by \( a, b \) and \( c \) respectively).

We define the linear transformation matrix \( T \) given below:

\[
T = \begin{bmatrix}
t_{11} & t_{12} & t_{13} \\
t_{21} & t_{22} & t_{23} \\
t_{31} & t_{32} & t_{33}
\end{bmatrix}
\]

Where \( T = [t_{11}, t_{12}, t_{13}] \) is the scheduling vector (in case of matrix multiplication is always \([1, 1, 1]\)) and \( S = \begin{bmatrix} T_2 \\ T_3 \end{bmatrix} = \begin{bmatrix} t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix} \) is transformation which maps \( P_{ind} \) into 2-dimensional systolic array.

**Definition 3:** Data dependency matrix is a matrix which consists of constant dependency vectors \( e_b, e_a, e_c \), each of them representing a data dependency corresponding to one of three variables \( (b, a, c) \).

Data dependency matrix for algorithm of matrix-matrix multiplication is given with:

\[
D = \begin{bmatrix}
e_b & e_a & e_c \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

The matrix \( T \) is associated with the so called projection direction \( u = [u_1, u_2, u_3]^T \) (there are some possible allowable projection vectors, see [1]), so that the following conditions must be satisfied:

1. \( \det T \neq 0 \) \hspace{1cm} (2)
2. \( T_i u = 0 \) and \( T_i u = 0 \) \hspace{1cm} (3)
3. \( \Delta_S = SD \) have to be from the set \( \{-1, 0, 1\} \). \hspace{1cm} (4)

The transformation matrix \( T \) maps the index point \( (i, j, k) \in P_{ind} \) into the point \( (t, x, y) \in T \cdot P_{ind} \) where:

\[
i = T_1[i \ j \ k]^T = i + j + k
\]
And for \((i, j, k) \in P_{\text{int}}\)
\[
[x \ y]^T = S[i \ j \ k]^T = \begin{bmatrix} t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix}^T
\]  
(6)

In this case \(t\) is time where calculations are performed, and \((x, y)\) are the coordinates of processors elements on 2-dimensional systolic array.

If \(P_{\text{in}} = \{P_{\text{in}}(a), P_{\text{in}}(b), P_{\text{in}}(c)\}\) is space of initial computations with:
\[
P_{\text{in}}(a) = \{(i,0,k)/1 \leq i, k \leq N\} \\
P_{\text{in}}(b) = \{(0, j,k)/1 \leq i, j \leq N\} \\
P_{\text{in}}(c) = \{(i, j,0)/1 \leq i, j \leq N\}
\]  
(7)

Then for the new position of the vector \(\gamma\), \(\gamma \in \{a, b, c\}\) is taken:
\[
p_{\gamma}^* = p_{\gamma} - (i + j + k - 2)e_{\gamma}
\]

Then
\[
\begin{bmatrix} x \\ y \end{bmatrix}_{\gamma}^* = S \cdot p_{\gamma}^* .
\]

Let us consider the case where \(u = [1 \ 1 \ 1]^T\).

From (3) we have:
\[
T_{u} = 0 \Rightarrow t_{21} + t_{22} + t_{23} = 0
\]  
(8)

\[
T_{u} = 0 \Rightarrow t_{31} + t_{32} + t_{33} = 0
\]  
(9)

Considering (1), (2), (4), (8) and (9), below are given all possible transformation matrices:
\[
\begin{bmatrix} 1 & 1 & 1 \\ \pm 1 & 0 & \mp 1 \\ \mp 1 & \pm 1 & 0 \end{bmatrix}, \quad \begin{bmatrix} 1 & 1 & 1 \\ \pm 1 & \mp 1 & 0 \\ \mp 1 & 0 & \pm 1 \end{bmatrix},
\]
\[
\begin{bmatrix} 1 & 1 & 1 \\ \mp 1 & \pm 1 & 0 \\ \mp 1 & \mp 1 & \pm 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 1 & 1 \\ 0 & \pm 1 & \pm 1 \\ \mp 1 & \mp 1 & 0 \end{bmatrix}
\]

To implement the mapping \((i, j, k) \xrightarrow{T} (t, x, y)\), first we define a linear mapping \(L = (L_1, L_2)\) such that \(P_{\text{ind}} \xrightarrow{L} P_{\text{ind}}^* \xrightarrow{T} P_{\text{ind}}^*\).

**Definition 4**: For algorithm \(1\), with the projection direction \(u = [1 \ 1 \ 1]^T\) the mapping \(L = (L_1, L_2)\) is defined in two possible cases, given with:
\[
L_1 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}, \quad L_2 = \begin{bmatrix} 0 \\ -1 \\ -1 \end{bmatrix}
\]  
(10)

or
\[
L_1 = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \quad L_2 = \begin{bmatrix} -1 \\ 0 \\ -1 \end{bmatrix}
\]  
(11)

Then the elements \((u,v,w) \in P_{\text{ind}}^*\) will be obtained from:
\[
\begin{bmatrix} u \\ v \\ w \end{bmatrix} = L_1 \begin{bmatrix} i \\ j \\ k \end{bmatrix} + L_2
\]

Let transformation matrix be:
\[
T = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix}
\]  
(12)

If we take the matrix \(L\) given with (10) we have:
\[
\begin{bmatrix} u \\ v \\ w \end{bmatrix} = L_1 \begin{bmatrix} i \\ j \\ k \end{bmatrix} + L_2 = \begin{bmatrix} i \\ i + j - 1 \\ i + k - 1 \end{bmatrix}
\]

From (6) for the new vector \((u,v,w)\) we can find the position of PE:
\[
\begin{bmatrix} x \\ y \end{bmatrix} = S \cdot \begin{bmatrix} i \\ i + j - 1 \\ i + k - 1 \end{bmatrix} = \begin{bmatrix} 1 - j \\ 1 - k \end{bmatrix}
\]

The new initial space will be:
\[
\hat{P}_{\text{in}}(a) = \{(i,0, i+k-1)/1 \leq i, k \leq n\} \\
\hat{P}_{\text{in}}(b) = \{(0, i+j-1, i+k-1)/1 \leq i, j, k \leq n\} \\
\hat{P}_{\text{in}}(c) = \{(i, i+j-1,0)/1 \leq i, j \leq n\}
\]  
(13)

Then we have:
Communication links are given with:

Now we can find the positions of input data in the array:

\[
\begin{align*}
&\alpha(i,0,i+k-1) & \rightarrow \begin{bmatrix} x \\ y \end{bmatrix} = S \cdot P_{\alpha}^*(a) = \begin{bmatrix} 3i+k-3 \\ 1-k \end{bmatrix} \\
&\beta(0,i+j-1,i+k-1) & \rightarrow \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 5-3i-2j-k \\ 5-3i-j-2k \end{bmatrix} \\
&\gamma(i,i+j-1,0) & \rightarrow \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1-j \\ 3i+j-3 \end{bmatrix}
\end{align*}
\]

(14)

(15)

(16)

For the coordinate system and for the data flow given with:

we have that \( b \) is flowing diagonally down, \( \alpha \) is flowing up and \( c \) to the left. The corresponding hexagonal array for \( N=4 \) is given in the fig. 1.

4. Systolic array for matrix multiplication without using of linear transformation matrix

On the other hand if we do not use transformation \( L \), and if we take the transformation matrix given with:

\[
T_i = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix}
\]

Then we will have:

\[
[x \ y]^T = S[i \ j \ k]^T = \begin{bmatrix} i-k \\ j-k \end{bmatrix},
\]

And:

\[
p_i^* = \begin{bmatrix} i \\ j \end{bmatrix}, \quad p_j^* = \begin{bmatrix} i \\ j \end{bmatrix}, \quad p_{k}^* = \begin{bmatrix} i \\ j \end{bmatrix}
\]

Now we can find the positions of input data in the array:

\[
\begin{align*}
&\alpha(i,0,k) & \rightarrow \begin{bmatrix} x \\ y \end{bmatrix} = S \cdot p_{\alpha}^* = \begin{bmatrix} i-k \\ 2-i-2k \end{bmatrix} \\
&\beta(0,j,k) & \rightarrow \begin{bmatrix} x \\ y \end{bmatrix} = S \cdot p_{\beta}^* = \begin{bmatrix} 2-j-2k \\ j-k \end{bmatrix} \\
&\gamma(i,j,0) & \rightarrow \begin{bmatrix} x \\ y \end{bmatrix} = S \cdot p_{\gamma}^* = \begin{bmatrix} 2i+j-2 \\ 2j+i-2 \end{bmatrix}
\end{align*}
\]

(17)

(18)
For the coordinate system and for the corresponding data flow which is like below:

We can conclude that \( b \) is flowing down, \( a \) to the right and \( c \) diagonally up. In the fig. 2 we show this array where \( N=4 \). (SHSA, and first was proposed by Kung and Leiserson [13]).

Now (without proof) we will give the theorem (taken from [7]) for 3-nested loop algorithm which is valid and for algorithm 1, taking on consideration that matrix multiplication is special case of 3-nested loop algorithm.

**Theorem 1** ([7]): The number of processors on the array for 3-nested loop algorithm with the size of loops \( (N_1, N_2, N_3) \) is given by:

\[
\Omega = \begin{cases} 
N_1N_2N_3 & \text{if } a_i > N_i \text{ for some } 1 \leq i \leq 3, \\
N_1N_2N_3 - \omega & \text{otherwise} 
\end{cases}
\]

(19)

where: \( \omega = (N_1 - a_1)(N_2 - a_2)(N_3 - a_3) \) and \( a_i = \left\lfloor \frac{T_{ii}}{\gcd(T_{11}, T_{12}, T_{13})} \right\rfloor \) where \( T_{ii}, i = 1,2,3 \) is (1,i) cofactor (minor) of matrix \( T \), and \( \gcd(T_{11}, T_{12}, T_{13}) \) is the greatest common divisor of \( T_{11}, T_{12} \) and \( T_{13} \).

Using the above theorem we have that the number of processors on SHSA array (which can be seen from fig.2 too) is:

\[
\Omega = N^3 - (N - 1)^3 = 3N^2 - 3N + 1
\]

In this case we have used that \( N_1 = N_2 = N_3 = N \) and for the transformation matrix given with (15) we have:

\[
|T_{11}| = |T_{12}| = |T_{13}| = 1
\]

from where \( a_1 = a_2 = a_3 = 1 \). Because we have that \( N=4 \) then \( \Omega = 37 \).

But we observed that the number of processor elements where we used the mapping \( L \) is 16 (fig. 1). Below is given the theorem associated with the proof which confirms this:

**Theorem 2** The number of processing elements in 2-dimensional systolic array for the algorithm of matrix-matrix multiplication (algorithm 1) for which is used the projection direction \( u = [1 \ 1 \ 1]^T \), could be reduced and given with \( \Omega = N^2 \).

**Proof:** We saw that for algorithm 1 with \( u = [1 \ 1 \ 1]^T \) could be applied the mapping \( L \) defined by (10). We take on consideration the linear transformation \( T \) in (1). The composition \( T \circ L \) was used for obtaining the corresponding systolic array, (because of the mapping \( P_{ind} \rightarrow P_{ind}^T \)). Therefore the number of processor elements \( \Omega \) depends on the matrix \( T \circ L \). But because in \( L = (L_1, L_2) \), the part \( L_2 = [0 \ -1 \ -1]^T \) contains no variables, we can conclude that \( \Omega \) depends only on composition \( H = T \circ L_1 \). Therefore we will have:

\[
H = T \circ L_1 = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \Rightarrow
\]

\[
H = \begin{bmatrix} t_{11} + t_{12} + t_{13} & t_{12} & t_{13} \\ t_{21} + t_{22} + t_{23} & t_{22} & t_{23} \\ t_{31} + t_{32} + t_{33} & t_{32} & t_{33} \end{bmatrix}
\]

And because of (8) and (9) we have that:

\[
H = \begin{bmatrix} t_{11} + t_{12} + t_{13} & t_{12} & t_{13} \\ 0 & t_{22} & t_{23} \\ 0 & t_{32} & t_{33} \end{bmatrix}
\]

Now we can see that cofactors \( H_{12} = 0 \) and \( H_{13} = 0 \). Using theorem 1 we have that \( a_2 = a_3 = 0 \) and \( a_1 = 1 \)

\[
a_i = \left\lfloor \frac{H_{11}}{\gcd(H_{11}, 0, 0)} \right\rfloor = 1.
\]

Therefore we will have (taking on consideration the fact that \( N_1 = N_2 = N_3 = N \))

\[
\Omega = N_1N_2N_3 - (N_1 - 1)N_2N_3 = N^3 - N^3 + N^2 = N^2
\]

Because of theorem 2, and taking on consideration that in construction of the array given in fig. 1 we are using the transformation \( L \).
given with (10), the number of processors (which can be seen from fig. 1) is \( \Omega = 4^2 = 16 \).

So, we showed (and mathematically proved) how the number of processors on the array can be reduced using the transformation \( L \). For \( N=4 \) is 16 vis-à-vis 37 without using the transformation.

On the below table we give the comparison for number of PE for different values of \( N \). This information is taken from [1], where can be find and some other very useful information’s and comparisons among different systolic arrays.

<table>
<thead>
<tr>
<th>( N )</th>
<th>Without using ( L )</th>
<th>By using ( L )</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>61</td>
<td>25</td>
</tr>
<tr>
<td>10</td>
<td>271</td>
<td>100</td>
</tr>
<tr>
<td>50</td>
<td>7351</td>
<td>2500</td>
</tr>
<tr>
<td>100</td>
<td>29701</td>
<td>10000</td>
</tr>
</tbody>
</table>

5. Conclusion

In this paper we have used the transformation matrix to optimize the parallel calculation of the problem of matrix-matrix multiplication. This algorithm has features given in definition 2, so it is very suitable for using systolic arrays as special case of processor network. We have used two types of linear transformation matrix for showing the result of our conclusion. We also have used two theorems to determine the number of processor elements on systolic arrays for matrix multiplication. We have emphasized the advantage of using the linear transformation \( L \), which implicates exactly on the reduction of the number of processing elements. We have also gave models of discussed systolic arrays.
Figure 2. The SHSA array for N=4 (without using the mapping L)

References


[16] www.cs.ucf.edu/courses/cot4810/fall04/presentations/Systolic_Arrays.ppt [01/06/2007]


[21] J.A.B. Fortes, K.S. Fu and B.W. Wah, Systematic design approaches for...


