# Analyzing of Some Performance Measures for Parallel Matrix Multiplication

Halil Snopce
South East Europian University,
CST Faculty, Tetovo 1200, R.
Macedonia
Email: h.snopce@seeu.edu.mk

Azir Aliu
South East Europian University,
CST Faculty, Tetovo 1200, R.
Macedonia
Email: azir.aliu@seeu.edu.mk

*Abstract*—**In order to make a proper selection for the given matrix-matrix multiplication operation and to decide which is the best suitable algorithm that generates a high throughput with a minimum time, a comparison analysis and a performance evaluation for some algorithms is carried out using the identical performance parameters**

***Keywords-** parallel algorithms for matrix multiplication, systolic array, linear transformation, nonlinear transformation, performance measures, number of processor elements Introduction.*

## I. INTRODUCTION

Most of the parallel algorithms for matrix multiplication use matrix decomposition that is based on the number of processors available. This includes the systolic algorithm [1], Cannon's algorithm [2], Fox's and Otto's Algorithm [3], PUMMA (Parallel Universal Matrix Multiplication) [4], SUMMA (Scalable Universal Matrix Multiplication) [5] and DIMMA (Distribution Independent Matrix Multiplication) [6]. The standard method for multiplying $n \times n$ matrices requires $O(n^3)$ multiplications. Most existing parallel algorithms are parallelization of the standard method. All implementations of the standard method have a cost, i.e., time-processor product of at least $O(n^3)$. Therefore, it is interesting to develop highly parallel and processor efficient algorithms that have less than $O(n^3)$ cost.

## II. WHY SYSTOLIC ARRAY?

The MPI technique needs two kinds of time to complete the multiplication process, $t_c$ and $t_f$, where $t_c$ represents the time it takes to communicate one data between processors and $t_f$ is the time needed to multiply or add elements of two matrices. It is assumed that matrices are of type $n \times n$. The other assumption is that the number of processors is $p$. Each processor holds $n^2/p$ elements and it was assumed that $n^2/p$ is set to a new variable $m^2$.

The number of arithmetic operations units will be denoted by $f$. The number of communication units will be denoted by $c$. The quotient $q = f/c$ will represent the average number of flops per communication access. The speedup is $S = q \cdot (t_f/t_c)$. We assume that the number of processors is $p = 4$. The dimension of the matrix is $n = 600$. The last assumption is that $t_f/t_c = 0.1$. Also we need to use the Efficiency formula $E = S/p$. In the table 1 we record the results that we obtain for all algorithms, under the assumptions that we made. [2, 3, 4, 7, 8, 9, 10, 11].

## III. DEFINITION OF SOME PERFORMANCE MEASURES FOR SYSTOLIC ARRAYS

**Definition 1:** The array size $(\Omega)$ is the number of PEs in the array.

**Definition 2:** The computation time $(T)$ is the sum of the time for the date input in the array-$T_{in}$, the time for the algorithm executing-$T_{exe}$ and the time necessary for dates leaving the array-$T_{out}$, i.e.

$$T = T_{in} + T_{exe} + T_{out} \tag{1}$$

**Definition 3:** The execution time, $T_{exe}$, is defined as:

$$T_{exe} = 1 + \max_{(t,x,y)\in P_{ind}} t - \min_{(t,x,y)\in P_{ind}} t_{out} \tag{2}$$

**Theorem 1:** [12] The execution time is given by the relation:

$$T_{exe} = 1 + \sum_{j=1}^{3} (N_j - 1) \cdot \left| \min_t t_{ij} \right| \tag{3}$$

**Definition 4:** The Pipelining period $(\alpha)$**:** The time interval between two successive computations in a PE. If $\lambda$ is a scheduling vector and $u$ is a projection direction, then the pipelining period is given by the relation:

$$\alpha = \lambda^T u \tag{4}$$

**Definition 5:** The geometric area $(g_a)$ of a two-dimensional systolic array is the area of the smallest convex polygon which bounds the PEs in the $(x, y)$-plane. The geometric area is given by the formula:

$$g_a = (N_1-1)(N_2-1)|T_{13}| + (N_1-1)(N_3-1)|T_{12}| + (N_2-1)(N_3-1)|T_{11}| \quad (5)$$

**Definition 6:**The Speedup $(S)$ of a systolic array is the ratio of the processing time in the SA to the processing time in a single processor $(T_1)$, i.e.

$$S = \frac{T_1}{T} \quad (6)$$

**Definition 7:** The Efficiency $(E)$ is defined as the ratio of the speedup to the number of PEs in the array i.e.

$$E = \frac{S}{\Omega} = \frac{T_1}{Tx\Omega} \quad (7)$$

**Theorem 2:** [12, 13]The number of processors on SHSA array is (the array where we have no using the linear transformation):

$$\Omega = 3N^2 - 3N + 1 \quad (8)$$

**Theorem 3:** [14] The number of processing elements in 2-dimensional systolic array for the algorithm of matrix-matrix multiplication for which is used the projection direction $u = [1 \ 1 \ 1]^T$, could be reduced and given with $\Omega = N^2$.

**Theorem 4:** [15] The number of PEs for the systolic array which is constructed by using the nonlinear transformation the number of PEs is given by the relation:

$$\Omega = n \cdot \left\lfloor \frac{3n-1}{2} \right\rfloor \quad (9)$$

**Definition 8:** 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 $P_{ind}$ is the set of index points and

$$Tt = T_1 [i \ \ j \ \ k]^T = i + j + k \quad (10)$$

## IV. The Advantage of Using the Linear Transformation in Designing the Systolic Array for Matrix Multiplication

From theorem 2, For N=4 then $\Omega = 37$ (which can be seen from fig.2 too). Because of theorem 3, the number of processors (which can be seen from fig. 1) is $\Omega = 4^2 = 16$. So, we can conclude how the number of processors on the array can be reduced using the linear transformation. For N=4 is 16 vis-à-vis 37 without using the transformation. On the table 3 we give the comparison for number of PE for different values of N. This information is taken from [13].

TABLE 3: COMPARISON FOR NUMBER OF PE

| N | Without using L | By using L |
|---|---|---|
| 5 | 61 | 25 |
| 10 | 271 | 100 |
| 50 | 7351 | 2500 |
| 100 | 29701 | 10000 |

For the pipeline period, in the case of the array in fig. 2, using relation (4) we get

$$\alpha = \lambda^T u = 3 \quad (11)$$

If this formula is used for the systolic array which is constructed by using the linear transformation matrix (the array in fig.1), we get $\alpha = 1$. This means that in the case of fig. 1 the PEs perform in every step.

If $k$ is an index point, then of course, $\max k = (N_1, N_2, N_3)$ and $\min k = (1,1,1)$. Using the relation (10) we have that:

$$\max_{(t,x,y)\in P_{ind}} t \ \ t = N_1 + N_2 + N_3; \ \ \min_{(t,x,y)\in P_{ind}} t \ \ = 1+1+1 = 3 \quad (12)$$

From relations (2) and (12) the execution time may is:

$$T_{exe} = 1 + N_1 + N_2 + N_3 - 3 = N_1 + N_2 + N_3 - 2 \quad (13)$$

If one uses the fact that if $N_1 = N_2 = N_3$, then $T_{exe} = 3N - 2$. On the other hand $T_{in} = T_{out} = N - 1$, so the computation time is:

$$T = 5N - 4 \quad (14)$$

In the case of the array in fig.1 the execution time will be ordered by using the theorem 1:

$$T_{exe} = 1 + (N_1 - 1)\cdot 0 + (N_2 - 1)\cdot 1 + (N_3 - 1)\cdot 1 = N_2 + N_3 - 1 \quad (15)$$

If $N_2 = N_3$ then $T_{exe} = 2N - 1$. In this case $T_{in} = N - 1$ and $T_{out} = 0$, therefore the computational time is

$$T = 3N - 2 \quad (16)$$

In the case of the array which was constructed by using the nonlinear transformation, we have that $T_{exe} = 3N - 1$, $T_{in} = N - 1$ and $T_{out} = 0$. Therefore the computation time is

$$T = 4N - 2 \quad (17)$$

For the geometric area in the case of the array in fig.2, if one takes $N_1 = N_2 = N_3 = N$ then

$$g_a = 3N^2 - 6N + 3 = 3(N-1) \tag{18}$$

For the second case (the array in fig. 1) the geometric area If one takes $N_1 = N_2 = N_3 = N$ will be calculated as

$$g_a = (N-1)^2 \tag{19}$$

In the case of the array with nonlinear transformation, one can calculate the geometric area in a similar way as above

$$g_a = 2(N-1)^2 \tag{20}$$

Since the duration of matrix multiplication on a system with only one processor is $T_1 = N^3$, the speedup and efficiency in the case of the array in fig.2, using relations (6) and (7), will be respectively:

$$S = \frac{N^3}{5N-4} \tag{21}$$

$$E = \frac{N^3}{(5N-4)(3N^2-3N+1)} \left( \lim_{N\to\infty} E = \frac{1}{15} = 6.7\% \right) \tag{22}$$

The same parameters in the case of using linear transformation matrix are:

$$S = \frac{N^3}{3N-2} \tag{23}$$

$$E = \frac{N}{(3N-2)} \left( \lim_{N\to\infty} E = \frac{1}{3} = 33.3\% \right) \tag{24}$$

And finally these parameters for the array where nonlinear transformation has been used are:

$$S = \frac{N^3}{4N-2} \tag{25}$$

$$E = \frac{N^2}{(4N-2)\cdot \left\lfloor \frac{3n-1}{2} \right\rfloor} \left( \lim_{N\to\infty} E = \frac{1}{6} = 16.7\% \right) \tag{26}$$

Using the results obtained by the relations (14-26), as well as theorems 1, 2, and 3, one can construct the corresponding table, where all the results can be compared. In table 2 it is given a comparison of performance characteristics for some values of $N$.

## V. Conclusion

In this paper are analyzed some performance measures for parallel matrix multiplication. We emphasized the systolic approach as most efficient. We can conclude that using the identical performance parameters, for each parameter, the array which is constructed using linear transformation matrix has better performances. Especially for the efficiency when $N$ tends to the infinity we have that it is approximately five times better than the array without using the linear transformation. From the table 2 we can deduce the advantage of using the linear transformation.

## References

[1] Choi, J., J.J. Dongarra and D.W. Walker, Level 3 BLAS for distributed memory concurrent computers, 1992. CNRS-NSF Workshop on Environments and Tools for Parallel Scientific Computing, Saint Hilaire du Touvet, France, Sept. 7-8, Elsevier Sci. Publishers.

[2] Alpatov, P., G. Baker, C. Edwards, J. Gunnels, G. Morrow, J. Overfelt, Robert Van de GEijn and J. Wu, 1997. Plapack: Parallel Linear Algebra Package, Proceedings of the SIAM Parallel Processing Conference.

[3] Agarwal, R.C., S.M. Balle, F.G. Gustavson, M. Joshi and P. Palkar, 1995. A 3-Dimensional Approach to Parallel Matrix Multiplication, IBM J. Res.Develop., Volume 39, Number 5, pp: 1-8, Sept.

[4] Choi, J., J.J. Dongarra and D.W. Walker, 1994. PUMMA: Parallel Universal Matrix Multiplication Algorithms on distributed memory concurrent computers, Concurrency: Practice and Experience, Vol 6(7): 543-570.

[5] Cannon, L.E., 1969. A Cellular Computer to Implement the Kalman Filter Algorithm, Ph.D. Thesis Montana State University.

[6] Chtchelkanova, A., J. Gunnels, G. Morrow, J. Overfelt, R. van de Geijn, 1995. Parallel Implementation of BLAS: Ceneral Techniques for Level 3 BLAS, TR-95-40, Department of Computer Sciences, University of Texas, OCT.

[7] Agarwal, R.C., F.G. Gustavson and M. Zuibar, 1994. A high-performance matrix multiplication algorithm on a distributed memory parallel computer using overlapped communication, IBM J. Res. Develop., Volume 38, Number 6.

[8] Ziad Alqadi and Amjad-Jazzar, 2005. Analysis of program methods used for optimizing matrix multiplication, J.Eng., Vol.15, NO. 1: 73-78.

[9] Anderson, E., Z. Bai, C. Bischof, J.Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, S. Hammarling, A. McKenney and D. Sorensen, 1990 Lapack: A Portable Linear Algebra Library for High Performance Computers, Proceeding of Supercomputing '90, IEEE Press, pp: 1-10.

[10] Edwards, C., P. Geng, A. Patra and R. Vande Geijn, 1995. Parallel matrix distributions: have we been doing it all wrong?, Tech. Report TR-95-40, Dept. of Computer Sciences, The University of Texas at Austin.

[11] Fox, G., S. Otto and A.Hey, 1987. Matrix Algorithms on a Hypercube I: matrix multiplication, Parallel Computing 3, pp:17-31.

[12] C.N. Zhang, J.H. Weston, Y. F. Yan: Determining object functions in systolic array designs. IEEE Trans. VLSI Systems 2, No. 3 (1994), 357-360.

[13] M.P. Bekakos, Highly Parallel Computations-Algorithms and Applications, Democritus University of Thrace, Greece, pp. 139-209, 2001.

[14] Snopce, H., Elmazi, L., Reducing the number of processors elements in systolic arrays for matrix multiplication using linear transformation matrix, Int. J. of Computers, Communications and Control, Vol. III (2008), Suppl. issue: Proceedings of ICCCC 2008, pp. 486-490.

[15] Gusev, M., and Evans, D.J., A new matrix vector Product Systolic Array, Parallel Algorithms and Applications, 22, 346-349, 1994.

## Appendix

TABLE 1: THEORETICAL RESULTS

| **Algorithm** | $f$ | $c$ | $q$ | **S** | **E** |
|---|---|---|---|---|---|
| Systolic algorithm | 55080000 | 1440000 | 38.25 | 3.825 | 0.956 |
| Cannon's algorithm | 271800000 | 108720000 | 2.5 | 0.25 | 0.0625 |
| Fox's algorithm with square decomposition | 162360000 | 270720000 | 0.599 | 0.0599 | 0.015 |
| Fox's algorithm with scattered decomposition | 54360000 | 109440000 | 0.497 | 0.0497 | 0.0124 |
| PUMMA | 54360000 | 1620000 | 33.55 | 3.355 | 0.839 |
| SUMMA | 54360000 | 1800000 | 30.2 | 3.02 | 0.755 |
| DIMMA | 54360000 | 1800000 | 30.2 | 3.02 | 0.755 |

TABLE 2: COMPARISON OF PERFORMANCE CHARACTERISTICS

| | Without using L | | | By using L | | | By nonlinear transf. | | |
|---|---|---|---|---|---|---|---|---|---|
| | N=4 | N=10 | N=100 | N=4 | N=10 | N=100 | N=4 | N=10 | N=100 |
| $\Omega$ | 37 | 271 | 29701 | 16 | 100 | 10000 | 20 | 140 | 14900 |
| $T$ | 16 | 46 | 496 | 10 | 28 | 298 | 14 | 38 | 398 |
| $S$ | 4 | 21.7 | 2016 | 6.4 | 35.7 | 3355 | 4.5 | 26.3 | 2512.6 |
| $E$ | 10.8% | 8% | 6.8% | 40% | 35.7% | 33.5% | 23% | 19% | 16.9% |
| $g_a$ | 27 | 243 | 29403 | 9 | 81 | 9801 | 18 | 162 | 19602 |

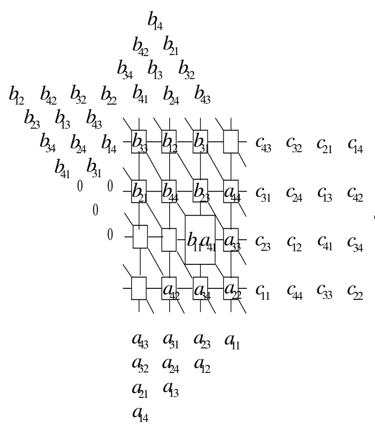
Fig. 1. Systolic array using theorem 3

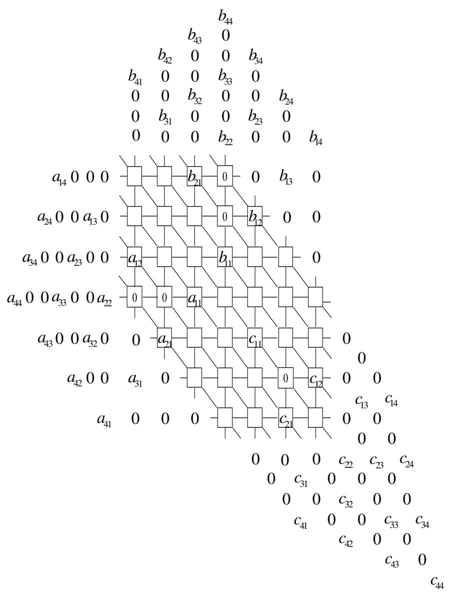
Fig. 2 The SHSA array for N=4

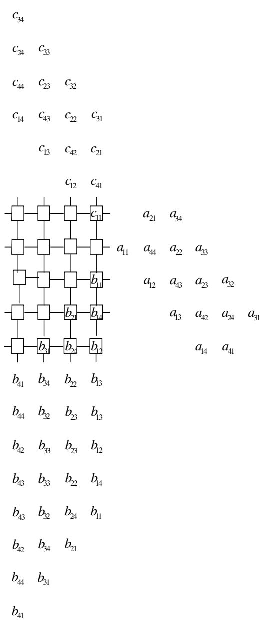
Fig. 3 the systolic with nonlinear mapping