An Efficient Double Pipe Systolic Array for Matrix Product

G. M. MEGSON
Computing Laboratory, University of Newcastle-Upon-Tyne, Claremont Tower, Claremont Road, Newcastle-Upon-Tyne NE1 7RU

An improved two layer systolic algorithm for computing the product of two \( n \times n \) band matrices \( A \) and \( B \) with bandwidths \( w_1 \) and \( w_2 \) respectively is described. The array requires \( 2\hat{w}_1\hat{w}_2 + \hat{w}_1 + \hat{w}_2 - 1 \) basic inner product cells where \( \hat{w}_1 = \lceil \frac{w_1}{2} \rceil \) and \( \hat{w}_2 = \lceil \frac{w_2}{2} \rceil \), has efficiency \( e = 1 \) and computation time \( T = 2n + (3/2)\min(\hat{w}_1, \hat{w}_2) \) inner product steps and requires only a single pass of data through the design.

Received September 1989

1. INTRODUCTION

In Ref. 3 a multiple layer systolic algorithm for evaluating the banded matrix product in the same time as the Kung & Leiserson hexagonal scheme, but with only half the processors and an efficiency of \( e = 0.66 \) rather than \( e = 0.33 \) was introduced. The design was volume efficient and consisted of two layers each containing a hexagonal array one quarter the size of the original which were coupled together by a bank of adders connected to the output edge of the arrays. In this paper we extend the above idea to produce a two-layer scheme which retains the cell reduction, but also improves the computation time and efficiency. The task is achieved in two stages as follows:

(i) The original double pipe computation is restructured to exploit the decoupling of partitioned matrix product computations while also reducing the effective bandwidth of sub-matrix computations.

(ii) The hexagonal array is re-timed to run slower than the original design and interleaving is applied to sub algorithms to improve efficiency.

Step (i) relies on a simple observation regarding the permutation of rows and columns of the matrices using the double pipe splitting of Ref. 3. Step (ii) is non-intuitive. Indeed the appeal of the design is that overall improvements in speed and efficiency can be produced by actually slowing down (not accelerating) the computations of component algorithms.

Now before proceeding some remarks are required to put the design in context. Throughout the discussion we assume that the reader is familiar with the data flow of previous designs (see Refs. 1–3) although for completeness the time area and efficiency of these schemes are given in Table 1. Area and Time are in terms of inner product step computations, and efficiency is the fraction of steps a cell is engaged in processing during the computation. The first two rows summarize Ref. 1 and 3. The entry in row 3 is the subject of this paper, and row 4 represents the most efficient hexagonal array known.  

\[
C = AB = (A_1 + A_2)(B_1 + B_2)
\]  

where

\[
A_1 = \begin{bmatrix}
 a_{11} & 0 & a_{13} & 0 & a_{15} & 0 \\
 0 & a_{22} & 0 & a_{24} & 0 & a_{26}
\end{bmatrix}
\]

\[
A_2 = \begin{bmatrix}
 0 & a_{12} & 0 & a_{14} & 0 & a_{16} \\
 a_{21} & 0 & a_{23} & 0 & a_{25} & 0 \\
 0 & a_{32} & 0 & a_{34} & 0 & a_{36} \\
 a_{41} & 0 & a_{43} & 0 & a_{45} & 0 \\
 0 & a_{52} & 0 & a_{54} & 0 & a_{56}
\end{bmatrix}
\]

\[
B_1 = \begin{bmatrix}
 b_{11} & b_{13} & 0 & b_{15} & 0 \\
 0 & b_{22} & 0 & b_{24} & 0 & b_{26} \\
 b_{31} & 0 & b_{33} & 0 & b_{35} & 0 \\
 0 & b_{42} & 0 & b_{44} & 0 & b_{46} \\
 b_{51} & 0 & b_{53} & 0 & b_{55} & 0 \\
 0 & b_{62} & 0 & b_{64} & 0 & b_{66}
\end{bmatrix}
\]

\[
B_2 = \begin{bmatrix}
 0 & b_{12} & 0 & b_{14} & 0 & b_{16} \\
 b_{21} & 0 & b_{23} & 0 & b_{25} & 0 \\
 0 & b_{32} & 0 & b_{34} & 0 & b_{36} \\
 b_{41} & 0 & b_{43} & 0 & b_{45} & 0 \\
 0 & b_{52} & 0 & b_{54} & 0 & b_{56} \\
 b_{61} & 0 & b_{63} & 0 & b_{65} & 0
\end{bmatrix}
\]

2. THE DOUBLE PIPE SPLITTING

Our first task is to restructure the double pipe computation. Recall that the double pipe splitting (Ref. 3) for two \( n \times n \) band matrices \( A \) and \( B \) of bandwidths \( w_1 \) and \( w_2 \) has the form

\[
C = C_1 + C_2 + C_3 + C_4
\]
Table 1. Area/Time requirements of various matrix product schemes

<table>
<thead>
<tr>
<th>Array</th>
<th>Time ((T))</th>
<th>Area ((A))</th>
<th>Efficiency (e)</th>
<th>(AT^2(n \gg w_1w_2))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kung &amp; Leiserson</td>
<td>(3n + \min(w_1, w_2))</td>
<td>(w_1w_2)</td>
<td>(e = 0.33)</td>
<td>(9n^2w_1w_2)</td>
</tr>
<tr>
<td>Megson &amp; Evans</td>
<td>(3n + \min(w_1, w_2) + 1)</td>
<td>(w_1w_2 + w_1 + w_2 - 1)</td>
<td>(e = 0.66)</td>
<td>(9n^2w_1w_2)</td>
</tr>
<tr>
<td>Megson</td>
<td>(2n + \min(w_1, w_2) + 1)</td>
<td>(w_1w_2 + w_1 + w_2 - 1)</td>
<td>(e = 1.0)</td>
<td>(4n^2w_1w_2)</td>
</tr>
<tr>
<td>Weiser &amp; Davis</td>
<td>(n + \min(w_1, w_2))</td>
<td>(w_1w_2)</td>
<td>(e = 1.0)</td>
<td>(n^2w_1w_2)</td>
</tr>
</tbody>
</table>

where

\[
C_1 + C_4 = A_1B_1 + A_2B_2 = \begin{bmatrix}
  c_{11} & 0 & c_{13} & 0 & c_{15} & 0 \\
  0 & c_{22} & 0 & c_{24} & 0 & c_{26} \\
  c_{31} & 0 & c_{33} & 0 & c_{35} & 0 \\
  0 & c_{42} & 0 & c_{44} & 0 & c_{46} \\
  c_{51} & 0 & c_{53} & 0 & c_{55} & 0 \\
  0 & c_{62} & 0 & c_{64} & 0 & c_{66}
\end{bmatrix}
\]

\(\hat{A}_1 = \begin{bmatrix}
  a_{11} & a_{13} & a_{15} & 0 & 0 & 0 \\
  a_{31} & a_{33} & a_{35} & 0 & 0 & 0 \\
  a_{51} & a_{53} & a_{55} & 0 & 0 & 0 \\
  0 & 0 & 0 & a_{22} & a_{24} & a_{26} \\
  0 & 0 & 0 & a_{42} & a_{44} & a_{46} \\
  0 & 0 & 0 & a_{62} & a_{64} & a_{66}
\end{bmatrix}\)

\(\hat{A}_2 = \begin{bmatrix}
  0 & 0 & 0 & a_{12} & a_{14} & a_{16} \\
  0 & 0 & 0 & a_{32} & a_{34} & a_{36} \\
  0 & 0 & 0 & a_{52} & a_{54} & a_{56} \\
  a_{21} & a_{23} & a_{25} & 0 & 0 & 0 \\
  a_{41} & a_{43} & a_{45} & 0 & 0 & 0 \\
  a_{61} & a_{63} & a_{65} & 0 & 0 & 0
\end{bmatrix}\)

\(\hat{B}_1 = \begin{bmatrix}
  b_{11} & b_{13} & b_{15} & 0 & 0 & 0 \\
  b_{31} & b_{33} & b_{35} & 0 & 0 & 0 \\
  b_{51} & b_{53} & b_{55} & 0 & 0 & 0 \\
  0 & 0 & 0 & b_{22} & b_{24} & b_{26} \\
  0 & 0 & 0 & b_{42} & b_{44} & b_{46} \\
  0 & 0 & 0 & b_{62} & b_{64} & b_{66}
\end{bmatrix}\)

\(\hat{B}_2 = \begin{bmatrix}
  b_{12} & b_{14} & b_{16} \\
  b_{32} & b_{34} & b_{36} \\
  b_{52} & b_{54} & b_{56} \\
  b_{21} & b_{23} & b_{25} & 0 & 0 & 0 \\
  b_{41} & b_{43} & b_{45} & 0 & 0 & 0 \\
  b_{61} & b_{63} & b_{65} & 0 & 0 & 0
\end{bmatrix}\)

such a splitting allows the matrices (2.5a) and (2.5b) to be computed on two hexagonal arrays with one quarter the size of the original hexagonal array. For example (2.5a) is constructed by computing \(C_1\) and \(C_4\) on separate arrays and adding the results emerging on the output connections. (2.5b) is computed similarly after the rows and columns of the input matrices have been permuted so that they match the input of (2.5a) in form.

A two layer design is now evident, we place each array on a separate level coupling them together on the output edge with a bank of adder cells. To complete the product (2.1) a multi-pass argument is adopted with computation of (2.5a) and (2.5b) pipelined behind one another. The time and efficiency indicated in row 2 of Table 1 is produced by utilising the zero structure of the matrices to re-time the hexagonal array computation so that only a single synchronising delay element is associated with every two real data elements and the total data length in a single pass is \(3n/2\).

Now, an alternative formulation of the same problem is produced if we apply permutations to the rows and columns of (2.2) and (2.3) to yield

\[
\hat{C} = (\hat{A}_1 + \hat{A}_2)(\hat{B}_1 + \hat{B}_2)
\]
where

\[
\begin{bmatrix}
C_{11} & C_{13} & C_{15} \\
C_{31} & C_{33} & C_{35} \\
C_{51} & C_{53} & C_{55}
\end{bmatrix}
\]

(2.7a)

\[
\begin{bmatrix}
C_{12} & C_{14} & C_{16} \\
C_{32} & C_{34} & C_{36} \\
C_{52} & C_{54} & C_{56}
\end{bmatrix}
\]

(2.7b)

\[
\hat{C}_1 = \hat{A}_1 \hat{B}_1 + \hat{A}_2 \hat{B}_2 = \hat{C}_2 = \hat{A}_1 \hat{B}_2 + \hat{A}_2 \hat{B}_1
\]

Hence,

\[
D = \hat{A}_{11} \hat{B}_{11} + \hat{A}_{12} \hat{B}_{21}, E = \hat{A}_{22} \hat{B}_{22} + \hat{A}_{21} \hat{B}_{12}
\]

(2.8a)

\[
F = \hat{A}_{11} \hat{B}_{12} + \hat{A}_{22} \hat{B}_{21}, G = \hat{A}_{21} \hat{B}_{11} + \hat{A}_{12} \hat{B}_{22}
\]

(2.8b)

where \(\hat{A}_i\) and \(\hat{B}_j\) are the obvious submatrices and \(D, E, F, G\) are the diagonal and anti-diagonal blocks of \(C\). Notice that if the original matrices are of bandwidths \(w_1\) and \(w_2\) the matrices in (2.8) have bandwidths \(\hat{w}_1 = \lceil w_1/2 \rceil\) and \(\hat{w}_2 = \lceil w_2/2 \rceil\) respectively. The practical value of this transformation is that a two layer design with the same characteristics, as the double pipe above, is produced without the need to re-time the reduced size array. To prove this statement simply compute \(\hat{A}_{11} \hat{B}_{11}\) and \(\hat{A}_{12} \hat{B}_{21}\) simultaneously on separate quarter sized hex arrays and add the results to produce \(D\). Now \(E, F, G\) can be computed in a similar manner apparently demanding four passes of data through the array. However the number of passes can be reduced to just two by interleaving the computations of \(D\) and \(E, F, G\) on the same passes, using the synchronising delays of the original data flow pattern. Now pass 1 produces \(\hat{C}_1\) and pass 2 \(\hat{C}_2\), the order of each of the matrices is \(n/2\), yielding the time for a pass as \(T = 3(n/2) + \min(\hat{w}_1, \hat{w}_2) + 1\). Thus with two sequential passes the result of Table 1 (row 2) is reproduced without re-timing.

3. A SLOW HEXAGONAL ARRAY

The efficiency of the scheme introduced above is \(e = 0.66\%\) because the interleaving fills one of the two synchronising spaces between data in the original method. Another design is easily produced by filling the remaining synchronising space of the first pass so that \(D, E, F\) are interleaved and \(G\) is computed on the second pass. The first pass then has \(e = 1\) and the second pass \(e = 0.33\) leaving the average efficiency \(e = 0.66\) and the computation time unchanged. Now, if the evaluation of \(G\) can be incorporated into the first pass, improved time as well as efficiency should result. Unfortunately there appears no obvious way to achieve the task because the interleaving of \(D, E, F\) makes the array cells compute continuously and forces \(G\) into another pass.

The problem is solved if an extra synchronising delay can be placed between the matrix elements input to the array, because then four problems can be interleaved on one pass. We conclude that the original hexagonal array must be re-timed to run more slowly. To perform this retiming consider the local correctness criteria shown in Fig. 1. Fig. 1(i) shows the connections of four basic cells in the standard hexagonal array, the circle denotes an active (currently computing) processor. The original data flow pattern in Ref. 1 is constructed simply by noting that the time to traverse the cycle from cell 1 to 3 is three steps. Each cell is active only once in three steps so each real data element has to be followed by two dummy synchronising values. To produce a design in which cells are active only once every four steps we re-time the loop as shown in Fig. 1(ii), by adding delays to the diagonal outputs of two cells (one in each loop). The dual problem is to add delays to the diagonal inputs of the cells instead of the outputs. Fig. 1(ii) demonstrates a modified hexagonal array for multiplying bi-diagonal matrices in \(T = 4n + \min(w_1, w_2)\) time with efficiency \(e = 0.25\) and \(w_1 = w_2 = 2\). It follows that cells are active one in four cycles and that three synchronising elements must follow every real data input. Arrays for higher bandwidths can be built up recursively by adding cells and applying the re-timing (or its dual) to groups of four cells, and this way we arrive at Fig. 2. A simple rule for general construction
is to add delays to all the diagonal outputs of even numbered rows of cells. Tracing out the snapshots as shown in Fig. 2, demonstrates that the design computes correctly, and that for the matrices to synchronise at most \((1/2)\min(\bar{w}_1, \bar{w}_2)\) steps must be added to allow the input matrices to filter through the additional delay cells and meet the result matrix \(C\) as required.

It remains only to solve the original matrix product (2.8) by using two of these re-timed hexagonal arrays, each with \(\bar{w}_1 \times \bar{w}_2\) cells. The computation of \(D\) proceeds by computing one matrix product term on each array and adding the results together using \(\bar{w}_1 + \bar{w}_2 - 1\) adder cells in \(T = 4(n/2) + (3/2)\min(\bar{w}_1, \bar{w}_2) + 1\) steps and efficiency \(e = 0.25\). Interleaving the computations of \(D, E, F,\) and \(G\) now requires just a single pass, and fills all the synchronising spaces, so that the area and time for computing (2.8) is the same as a single pass but has efficiency \(e = 1.0\). The entry in row 3 of Table 1 results immediately when we assume that the delays represent negligible area compared with an inner product cell (and can be subsumed into even row cells as extra registers).

4. CONCLUSION

We have developed a new double pipe array which computes a matrix product approximately one third faster than the original hexagonal array, and twice slower than the design of Weiser and Davis. However, the new array is just as efficient as the latter and uses approximately half the hardware of either scheme.

The last column of Table 1 grades the new scheme with respect to the area time bound \(AT^2\) in Ref. 6, and we see clearly that under this criteria double pipe schemes improve the performance of the matrix product problem but that the Weiser-Davis array retains the best trade-off. Furthermore, it is fairly obvious that the splitting (2.7) can be applied directly to the Weiser-Davis array to produce a double pipe array. Simply use two quarter sized Weiser-Davis arrays and a bank of adders to compute (2.8) in four passes, and produce a time of \(T = 2n + \min(\bar{w}_1, \bar{w}_2) + 1\) using the fact that \(D, E, F,\) and \(G\) are of order \(n/2\). Notice that this array is \((1/2)\min(\bar{w}_1, \bar{w}_2)\) cycles faster than the double pipe array just derived. Both of the arrays are 100% efficient and the number of passes cannot be reduced, we conclude that no further improvements are possible unless the number of products in (2.8) is reduced, or we allow data to be loaded into cells and remain station-
ANNOUNCEMENTS

27-30 AUGUST 1990
AMSTERDAM
EUROMICRO 90

Sixteenth Symposium on Microprocessing and Microprogramming

Hardware and Software in System Engineering

Euromicro 90 is the sixteenth annual Symposium organised by Euromicro. The purpose of the Symposium is to bring together people from business, industry, research, government and academia, who are interested in all problems related to the underlying technology and use of microprocessor systems.

The emphasis of the Euromicro 90 Symposium will be the hardware/software interface and system reliability.

The theme of Euromicro 90 is Hardware and Software in System Engineering with the sub-themes of The Moving Hardware/Software Interface and System Reliability.

Programme

The EUROMICRO Symposium is a major annual event around which a collection of related events congregate.

EUROMICRO 90 will include a mainstream programme based on Keynote Sessions, Scientific Paper Sessions, Short Note Sessions and a special Session devoted to Computer Music.

Tutorials on hot topics will be provided. ESPRIT groups will be invited to provide joint presentations and Social Events will be organised, offering additional opportunities for meetings with people from countries all over the world.

ESPRIT

EUROMICRO membership and Symposium attendance is wider than the European Community. However the EUROMICRO Symposium does provide a focal point for ESPRIT, BRITE and similar project partners to meet in order to disseminate their results to a wider audience.

For further information contact:
EUROMICRO, Attn. Mrs Chiquita Snipe-Marlisa, P.O. Box 545, 7500 AM Enschede, The Netherlands. Tel: +31 53 337899. Fax: +31 53 337415. Telex: 44200 thes

25-27 SEPTEMBER 1990
NOTTINGHAM UNIVERSITY

The NCUF Conference 1990

Open Systems in Business

The National Computer Users Forum is holding its 4th annual conference at Nottingham University from Tuesday, 25th September to Thursday, 27th September, 1990.

There are plenary sessions for all the delegates on each of the days of the conference. There will also be two parallel streams focusing on Interworking and the Applications Environment.

The programme also has workshop and tutorial sessions during which case studies and topics of a specialised nature are presented. Numbers in these sessions are limited to 25 delegates so that they can discuss issues of specific interest in detail with expert workshop leaders.

It is anticipated that there will be in excess of 150 delegates. The conference will consider topical issues arising from applications portability and communications networking in an Open Systems environment, a proprietary environment and in a mixed environment.

For further information contact:
The Conference Secretary, The NCUF Conference, PO Box 18, Beeston, Nottingham NG9 3LT. Fax: 0602-280823

26-28 SEPTEMBER 1990
QUEEN ELIZABETH II CONFERENCE CENTRE, LONDON

International Conference on System Development Environments and Factories

The International Conference on System Development Environments and Factories was jointly organised for the first time in 1989 by major European and American Research and Development Projects. The great success of the first conference gave rise to the establishment of the conference as a regular event that is aimed at providing a forum for the presentation of results achieved in those projects, for discussions between providers and potential users of environments and factories and for the demonstration of systems.

The second conference will focus on

Adopting Environment and Factory Concepts in Industry

by presenting information on:
- Technical and Scientific Advances
- Current Status of Projects in Europe, the USA and Japan
- Financial, Human and Organisational Impact on Environments and Factories on Industry
- Case Studies Conducted in Industry
- Government Initiatives

For further information contact:
SDE & F OFFICE, Sema Group, 143/149 Farrington Road, London EC1R 3AD. Tel: +44(0)1-831 6144. Fax: +44(0)1-278 0574

REFERENCES