A Systolic Array Graph Partitioning System

Brian J. d'Auriol[1]

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada, R3T 2N2

Meera B. Dugar

Department of Computer Science and Engineering,
Wright State University,
Dayton, Ohio,
USA, 45435

Abstract

An X-windows based graphical interface system called the Systolic Array Graph Partitioning system is proposed. This system allows for the partitioning of a given systolic array in the context of a specific model for compiling systolic computations for multicomputers. Particular features of this system include the determination of the topology of the multicomputer and an analysis of the mapping of computations to processors.

Keywords: Systolic Array, Partitioning, Multicomputer, Load Balancing.

1 Introduction

The partitioning problem for systolic arrays has received much attention in the literature. See for example the discussion of Moldovan in [1, Chapter 5]. However, most of the discussion pertains to *computational issues*, often in context of VLSI implementations. For example, a given systolic array solution to a problem may require more systolic processing cells than is available on the targetted hardware device. Thus the systolic array solution requires partitioning so that the full computation can be carried out. Also, much of the analysis of systolic arrays have reflected VLSI fabrication issues (e.g. area, time and I/O pin counts). Little work has been done on the *effect* of the partitioning and subsequent mappings of regions of the systolic array to MIMD processors. That is to say, the partitioning of a particular systolic array results into associated machine topologies when a full mapping of the systolic computations and related communication is made to the physical machine. A detailed examination of this mapping process appears in [2, 3, 4]. A brief review of this model is presented herein for the convenience of the reader.

In a sense, the known methods of determining systolic arrays from data dependency graphs constitute an intent similar to that considered in this paper. The processing requirements in our case are different and often more stringent than those associated with the determination of systolic arrays. For example, the resulting connectivity of the target processors due to the partitioning and mapping process we describe in this paper significantly impacts upon the high level program representation of the implementation. As such, program representation is of primary concern to us, we are very much interested in the effect different partitions and mappings have on the connectivity of a MIMD computer. Furthermore, it is known that straightforward implementations of systolic arrays (without due consideration of partitioning) result in program representation which, when compiled and executed, do not execute efficiently, often resulting in slow-down. This is due to the mismatch in granularity. Since efficiency is also of significant concern to us, we are likewise interested in the granularity analysis.

Consequently, it is of particular importance to consider the physical processor topology in an unconstrained way that is derived from the partitioning process. Information that we may be interested in includes the numbers of systolic array cells allocated to a single processor and the numbers of corresponding communication links allocated to inter-processor communication links (i.e. the granularity issue), overall processor connectivity (e.g. linear, unidirectional, bidirectional, etc.) and, the internal orientation and connectivity of systolic cells which are allocated to the same processor.

This paper presents an X-windows based graphical interface system called the Systolic Array Graph Partitioning System which allows for the partitioning of a given systolic array in the context of the unified model. It has, however, possible utility in work not relating to the unified model. This partitioning system is aimed at providing the following features:

- visualization and manipulation of pertinent geometric systolic array parameters,
- determination and visualization of various spatial partitions of the systolic array, and
- determination, visualization and analysis of the resulting physical processor topology due to the suggested spatial partitions.

[1]This work was done while the author was at Dept. of Comp. Sci, Wright State University, Dayton, Ohio, USA, 45435.

Some previous work on partitioning of systolic arrays as well as background information regarding the unified model is given in Section 2. An overview of the Systolic Array Graph Partitioning System is given in Section 3. Concluding remarks are given in Section 4.

2 Related Work

A survey of some previous developments in partitioning techniques is given in [5]. The authors discuss the two principal forms that partitioning may take — LPGS (locally parallel globally sequential) or LSGP (locally sequential globally parallel). As summarized in [5], the LPGS form has been studied by Moldovan and consists of partitioning the systolic array into blocks which are then time scheduled one after the other on the physical computing platform. The LSGP form allocates each partition block to a physical processor. Thus, the LSGP form performs a spatial reduction of the original systolic array. The partitionings considered by the system proposed in this paper are of LSGP type.

As part of the unified model for compiling systolic computations for multicomputers [2, 3, 4], the authors propose a partitioning algorithm which takes a hexagonal wavefront array (an example of which is given in Figure 1) and a set of partitioning curves (illustrated by heavy dashed lines in Figure 1) to generate a processor graph representing the required machine topology. The algorithm is repeated in Figure 2 for the convenience of the reader; the validaty of the result is guaranteed by theorum 1 in [3]. The geometry of an arbitrary systolic array is denoted by $G = (V, E, \sigma_-, \sigma_+)$, where V is the set of vertices, E is the set of directed edges, and σ_-, σ_+ are functions which map E to V with $\sigma_-(e), \sigma_+(e)$ representing the source and destination nodes respectively for the edge $e \in E$. The processor topology of the target machine is denoted by $G^p = (V^p, E^p, \sigma_-^p, \sigma_+^p)$. This graph consists of supernodes and superedges. The set of partition curves is denoted by $\mathbf{C}$. Lastly, a special set of edges not existing in G is denoted by $\bar{E}$. As part of the work reported in [2, 3, 4], the systolic array is embedded into the Cartesian co-ordinate system thus given V and E identification and allowing intersection calculation of curves in $\mathbf{C}$ with each edge in the systolic graph: the direction of the x and y axis is given in the upper left portion of Figure 1 and the origin is shown as the filled in node.

3 Overview

In order to meet the objectives set forth in Section 1, the Systolic Array Graph Partitioning System incorporates a two phase interface. The first phase allows the user to define a systolic array by adjusting one or more *geometrical* properties of the systolic array. The user may then experiment by setting various types of partitions.

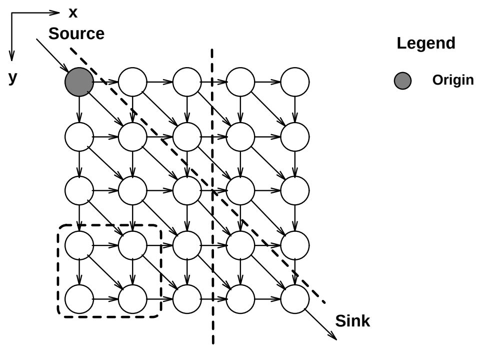

Figure 1: An example systolic array which is to be partitioned.

```
Choose $\mathbf{C} \in \zeta$.
Let $G^p$ be an empty graph initially.
For each edge, $e \in E \cup \bar{E}$ do
  BEGIN

  There are two possibilities: either the edge intersects with a $c \in \mathbf{C}$
  or it does not.

    1. If an $e$ intersects with $c$ then
       BEGIN
         (a) if $\sigma_-(e)$ is not already part of the graph of a supernode,
             then
             $V^p \leftarrow V^p \cup \sigma_-(e)$,
         (b) if $\sigma_+(e)$ is not already part of the graph of a supernode
             then
             $V^p \leftarrow V^p \cup \sigma_+(e)$,
         (c) add $e$ to the superedge $E^p$.
       END
    2. Otherwise
       BEGIN
         (a) if $\sigma_-(e)$ is in a graph and if $\sigma_+(e)$ is not in a graph of a
             supernode, then
             add $\sigma_+(e)$ to the graph of a supernode containing $\sigma_-(e)$.
             Otherwise, if neither $\sigma_-(e)$ nor $\sigma_+(e)$ is an any graph
             of a supernode in $V^p$, then add them both to the same
             supernode.
       END

END
```

Figure 2: Partitioning algorithm.

Currently only three partition strategies have been incorporated into the system: Diagonal cuts, cuts orthogonal to the x-axis and cuts orthogonal to the y-axis. Also, the placement of the cuts is automatically done by the system. Once the user has set a particular partitioning strategy for a specific geometry of the systolic array, the user can then apply the partitions and generate the associated physical processor topology required to implement that partitioning strategy. (Planned enhancements to the system allow for increased flexibilitiy.)

Two examples of the currently existing version of the partitioning system are shown in Figures 3(a) and 3(b). Figure 3(a) illustrates a two-curve partitiong of the systolic array along curves defined to be perpindicular to the wavefront in the systolic array. The resulting processor topology requires three processors, each interconnected as shown by bidirectional communication links.

The allocation of the source and sink to the same processor suggests simple connectivity between a host and the multicomputer (as in, for example, a backend computing engine as frequently found in multicomputer configurations). Figure 3(b) illustrates a four-curve partitioning of the systolic array along the horizontal. The resulting processor topology, substantively different than that of the preceding partitioning example, is a linear pipeline consisting of five processors. In this case, although the source and sink are allocated to different processing nodes, a ring topology implementation easily accomodates the processing requirements of a single host to multicomputer connection.

4 Conclusion

In this paper, we have proposed an X-windows based graphical system to assist in the studying of systolic array partitioning strategies. The system, called the Systolic Graph Partitioning System, allows for visualization and analysis of different systolic array partitioning strategies. We expect that when enhancements to the system are completed shortly, the resulting system will be able to better assist us with regards to identifying potential useful partitioning strategies for implementation of systolic computations on multicomputers.

References

[1] D.I. Moldovan, *Parallel Processing, From Applications to Systems.* 2929 Campus Drive, Suite 260, San Mateo, CA, USA, 94403: Morgan Kaufmann Publishers Inc., 1993.

[2] B.J. d'Auriol and V.C. Bhavsar, "Multicomputer Implementations of Systolic Computations: A Unified Approach," *Proc. of the 10th Annual International Conference on High Performance Computers (HPCS'96)*, Ottawa, Ontario, Canada, June, 5-7, 1996, June 1996.

[3] B.J. d'Auriol and V.C. Bhavsar, "A Unified Approach for Implementing Systolic Computations on Distributed Memory Multicomputers," Tech. Rep. WSU-CS-95-02, Department of Computer Science and Engineering, Wright State University, Dayton, Ohio 45435, USA, December 1995.

[4] B.J. d'Auriol, *A Unified Model for Compiling Systolic Computations for Distributed Memory Multicomputers.* PhD thesis, Faculty of Computer Science, University of New Brunswick, Fredericton, N.B. Canada, June 1995.

[5] A. Darte, T. Risset and Y. Robert, "Synthesizing Systolic Arrays: Some Recent Developments," *Proceedings of the International Conference on Application Specific Array Processors (Cat. No.91TH0382-2)*, Barcelona, Spain, Sept., 2-4, 1991, Los Alamitos, CA, USA: IEEE Compute. Soc. Press 1991, pp. 372–386, Sept. 1991.

Figure 3: The Systolic Array Graph Partitioning System: (a) (top) two-curve partitioning along diagonal, and (b) (bottom) four-curve partitioning along horizontal.