Journal of Computing and Information Science in Engineering

Book Review

"Highly Parallel Computations: Algorithms and Applications," edited by M. P. Bekakos, WIT Press, 2001

C. Evangelinos, Design Laboratory, Ocean Engineering Department, MIT.

Introduction

The volume is part of a series on advances in High Performance Computing with contributors and editors coming from a varied international background. Previous titles in the series concentrated mainly on high performance engineering applications.

The book is organized in three parts: The first three chapters address issues in the general area of parallel/grid computing. The next seven chapters deal with various algorithms for systolic arrays. The final two chapters cover algorithms and applications for neural networks. With such a wide range of subjects covered and with most chapters requiring some domain-specific background, there is bound to be a concern that the focus of the book is diluted. Concentrating on just systolic arrays would have made the volume an excellent background reference. The other two sections on their own do not cover enough material to satisfy their respective target audiences and would be better served by being part of other, domain-specific, volumes. Nevertheless, the contributions chosen for the volume are very interesting and the presentation of the material is of good quality. The major parts of the book are reviewed below.

General Purpose Parallel Computing

The 1st chapter, by D.J. Evans and M.Y. Saman, is a concise introduction to parallel computing concepts, parallel hardware and software models and programming paradigms with a tutorial introduction to MPI, employing examples that range from the simple (the classical master-slave calculation of π) to the more involved (a solution of the ''Torsion problem,'' a Poisson equation in 2D with Dirichlet boundary conditions using a finite difference scheme and a Jacobi iterative solver). Complete working code is provided in the text, written in simple, C-like C++ (which should allow Fortran programmers to understand the code). Standard blocking point to point communications are employed as well as some collective communications and Cartesian virtual topologies are also introduced in the case of the Torsion problem. The reviewer feels however that the master-slave parallel paradigm, which is employed in three out of the four examples is covered at the expense of more memory and communication efficient approaches.

The 2nd chapter, by P. Tsanakas et al., discusses scheduling techniques for UET-UCT (Unit Execution Time-Unit Communication Time) grid topologies, for both bounded and unbounded number of processors. The authors begin by showing how such n-dimensional grid topologies represent the iteration space of nested loops with dependencies. They then proceed to discuss basic concepts in scheduling for grids (precedence constraints, validity, makespan, processor-span, optimality etc.) and follow that with scheduling algorithms for optimal UET and UET-UCT grids. The exposition is theoretical, structured as a succession of definitions, theorems and examples. Closed form optimal solutions for makespan (time) and processor-span are derived. Several figures are provided to clarify a presentation that requires careful reading.

The 3rd chapter, by T.G. Typou and K.G. Margaritis, is a review of the current state of meta computing or Grid Computing. As such, it is unfortunately bound to become outdated very quickly with the multitude of recent developments in Grid-related projects. The first half of the chapter is a well-written introduction to metacomputing concepts that would be valuable as background reading for a student or researcher new to the field. The second half of the article concentrates on five specific meta-computing projects.

Systolic Arrays

The 4th chapter, by S. Sedukhin and S. Peng, introduces various approaches to designing Systolic Array Processors (SAP) for the n-dimensional Discrete Fourier Transform (DFT). For the 2D case, a scheme based on matrix multiplication is first described, followed by an enhancement using Horner's rule that avoids the precomputation of all factors and finally by a further modified third scheme. In each case, a systematic approach based on the 3D data dependency graph allows for the construction of an optimal SAP. For higher dimensional DFTs however, a directional splitting technique is proposed, decomposing the operation to a sequence of 1D DFTs, transforming the problem to one of an optimal design of building block Linear Array Processors (LAPs).

The 5th chapter, by D.J. Evans and C.R. Wang, describes a large number of systolic array designs for dense and banded matrix multiplications. The Regular Iterative Algorithm (RIA) is explained and then applied in detail to produce a total of 19 different designs. Figures explaining every design are provided to help the reader. A description of the possible measures of systolic array performance (size, time, speedup, efficiency etc.) is followed by a comparison based on these measures of the different designs.

The 6th chapter, by M.P. Bekakos et.al., discusses hexagonal systolic array design for matrix-matrix multiplication. A standard design methodology for the space-time mapping of nested loop algorithms based on the data dependency graph is first introduced. Then a modified procedure is introduced that allows for the generation of optimal arrays according to a given performance measure. This is followed by a survey of published hexagonal array designs, presenting four different designs, and comparing them in terms of size, time, speedup and efficiency and geometric and chip area. A modification of the optimal design is then proposed, to reduce the execution time at the expense of more processing elements. Another modified approach is proposed to produce systolic array designs for fault tolerance (that perform redundant computations using majority voting for the final result) or for higher throughput (performing multiple independent calculations concurrently).

The 7th chapter, by G. Oka and M. Vajteric, discusses the design of a systolic array algorithm for the singular value decomposition (SVD) of a matrix using the two-sided block Jacobi method.

The underlying topology is that of a 2D torus (toroidal mesh). The algorithm consists of iterations of four phases, with the first one involving communications between PEs and the other three computations and memory operations on systolic subarrays local to individual PEs.

The 8th chapter, by P. Capello, Eeciolu and C. Scheiman, describes a general technique for extracting a lower bound on the number of PEs required to achieve a given schedule for a regular array calculation or a system of uniform recurrence relations. Both can be represented as a Directed Acyclical Graph (DAG) with the graph edges corresponding to data dependencies. The DAG nodes can be seen as lattice points in a convex polyhedron. An augmented Diophantine system of equations is generated by employing the linear constraint of the schedule and the number of solutions to the system provides the required bound. Detailed examples are given for matrix product, Gaussian elimination, transitive closure and tensor product, providing in each case a time-optimal schedule that conforms to the computed lower bound.

The 9th chapter, by Th. Lippert and N. Petkov, discusses hyper-systolic algorithms and some of their applications. It is the largest in the book as it also provides a concise and useful review of systolic algorithms in order to introduce hyper-systolic algorithms as an extension to systolic ones. As part of this review several classic algorithms are presented and various mappings from a size N systolic array to a size P 1D processor ring. Hyper-systolic algorithms are introduced as generalizations of systolic arrays on 1D rings allowing for storage of intermediate results and shifts along the ring with non-unit strides. As a result hyper-systolic algorithms have a lower communication complexity and perform far better on parallel machines. Specific hyper-systolic implementations of the N-body problem (found at the heart of many physical problems) are presented, with explanatory figures and experimental results from a Cray T3D. Similarly a hyper-systolic algorithm is presented for the distributed matrix-matrix product, contrasted with its systolic counterpart and both are experimentally compared on a Myrinet-connected Alpha-Linux cluster.

The 10th chapter, by O.B. Efremides and M.P. Bekakos, discusses a new approach to the construction of optimal time-processor systolic arrays. While based on the data dependency graphs like the standard methods, it allows for nonlinear transformations to be used, providing for the generation of efficient systolic arrays in space and time. This is demonstrated for the cases of matrix-vector and matrix-matrix products.

Neural Networks

The 11th chapter, by H.Y.Y. Sanossian, provides both a review of the available methods for automated word recognition and also presents a new segmentation algorithm and feature extraction technique for Arabic characters. Both printed as well as handwritten Arabic is mainly cursive, thus presenting special difficulties. The first part of the chapter discusses the three stages of optical word recognition: segmentation, feature extraction of the sub-images and final classification. Four categories of algorithms are discussed for segmentation: straight, recognition-based, cut-classification and holistic. Three classes of feature extraction methods are also described: moments-, topological features- and neural network-based ones. Then a particular algorithm employing recognition-based segmentation, topological feature extraction and a neural network classification stage is presented, designed with the specific issues of cursive Arabic characters in mind. Feedback from the classification to the segmentation stage is also introduced to reduce errors, providing a more than 90% success rate.

The final chapter, by S.D. Likothanassis and E.F. Georgopoulos, discusses three new training algorithms for self-organized neural networks. Initially a Modified Genetic Algorithm (MGA) is introduced. It is used to evolve a population of multi-layered perceptron (MLP) neural networks to a near optimal network architecture at the same time as training them. Then a different approach is suggested, treating a neural unit as a p-input, one output non-linear system and employing the Linear Extended Kalman Filter (LEKF) as the training algorithm. A generalization of this approach is then proposed, with the evolution performed in the hidden region. Extensive results from the application of these algorithms in the cases of different real (as well as simulated) data are presented.