1988

Parallel (//) ELLPACK: An Expert System for Parallel Processing of Partial Differential Equations

Elias N. Houstis
Purdue University, enh@cs.purdue.edu

John R. Rice
Purdue University, jrr@cs.purdue.edu

T. S. Papatheodorou

Report Number:
88-831

PARALLEL (//) ELLPACK: AN EXPERT SYSTEM FOR PARALLEL PROCESSING OF PARTIAL DIFFERENTIAL EQUATIONS

E. N. Houstis
J. R. Rice
T. S. Papatheodorou

CSD-TR-831
November 1988

PARALLEL (//) ELLPACK: AN EXPERT SYSTEM FOR PARALLEL PROCESSING OF PARTIAL DIFFERENTIAL EQUATIONS

E.N. Houstis† and J.R. Rice**

Department of Computer Science
Purdue University
West Lafayette, IN 47907

T.S. Papatheodorou†

University of Patras
Department of Computer Engineering
and
Computer Technology Institute
Patras, Greece

Technical Report CSD-TR-831
CAPO Report CER-88-43
November, 1988

Abstract

We are developing an expert system environment for solving elliptic partial differential equations(PDEs) defined on two and three dimensional domains on MIMD type parallel machines. According to its design objectives, it will provide a uniform programming environment for implementing parallel MIMD PDE solution solvers, an automatic partitioning and allocation of the PDE computation, a very high level problem specification language, an interactive high level environment for grid selection, a domain partitioning and mapping facility, a uniform environment for obtaining software engineering measurements and a graphical display of solution output.

The //ELLPACK is implemented on a hardware facility consisting of a graphics workstation supporting X11 window system and connected to an NCUBE and SEQUENT machines through a wide bandwidth local network. The software infrastructure includes i) a PDE problem oriented language processor, ii) a geometry processing tool which is capable of generating fixed meshes and domain decompositions automatically and interactively, iii) an algorithm mapper facility for partitioning and mapping the underlying PDE computation and iv) an expert front end that selects the discretization mesh, the parallel algorithm/machine pair and its configuration. In order to support the solution of elliptic PDEs on fixed meshes, we are building a library of finite difference and element discretization methods and direct/iterative solvers for the NCUBE and SEQUENT machines.

* Research supported in part by the Air Force Office of Scientific Research under grant 84-0385 and the Strategic Defense Initiative under Army Research contract DAAL03-86-K-01606.
† Research supported in part by ESPRIT project 1588.

I. INTRODUCTION

In the next five to ten years we will see the widespread use of computer facilities consisting of layers of hardware systems organized with respect to their capabilities. These facilities will include powerful graphics workstations, parallel MIMD systems with tens of processors in the Billion Instructions Per Second (BIPS) class, parallel MIMD systems with hundreds of processors in the k Million Instructions Per Second (kMIPS) class, and special parallel archirectures suitable for numeric and symbolic processing (SPAN). This new hardware technology has widened the gap between hardware and applications. The recently established laboratory for Computing About Physical Objects (CAPO) at Purdue University and the SPAN group at the Computer Technology Institute of Patras-Greece, have established research projects that study the problem and are developing a software/algorthmic infrastructure that tries to close this gap, at least for certain applications which are governed by Partial Differential Equations (PDEs). Specifically, we try to address the following fundamental research questions:

> *How does one create a system where parallel PDE algorithms can be designed and specified in a reasonable time and which provides good implementation mappings to the future computing facilities?*

Our work concentrates on the following components of the above fundamental problem:

1. Develop a parallel PDE solving system for future computer facilities.
2. Develop and analyze partitioning and allocation strategies for PDE computations.
3. Create, analyze and implement parallel PDE solvers.
4. Evaluate new parallel languages and architectures which allow an efficient integration of numeric and symbolic processing.

The above components of research are described in more detail in the following sections. In this section we present a brief description of two tasks of the parallel ELLPACK (//ELLPACK) project.

I.A The Parallel ELLPACK Problem Solving System

We are developing an expert system environment for solving elliptic partial differential equations defined on two and three dimensional domains using MIMD machines. Its design objective is to provide a uniform programming environment for implementing parallel MIMD PDE solvers, automatic partitioning and allocation of the PDE computation, a very high level problem specification language, an interactive high level environment for grid selection, a domain partitioning and mapping facility, a uniform environment for obtaining software engineering measurements, and a graphical display of solution output. Two implementations of the parallel ELLPACK system are under development. The CAPO //ELLPACK system is implemented on a hardware facility consisting of a graphics workstation supporting X11 window system and

connected to NCUBE and SEQUENT machines through a wide bandwidth local network. The CTI-SPAN //ELLPACK prototype is implemented in the SPAN languages SPINLOG and PARLE and a virtual machine(VMC) that supports them [McCa 87a,87b, Simo 87]. The software infrastructure of //ELLPACK includes i) a PDE problem oriented language processor, ii) a geometry processing tool which is capable of generating fixed meshes and domain decompositions both automatically and interactively, iii) an algorithm mapper facility for partitioning and mapping the underlying PDE computation, and iv) an expert front end that selects the discretization mesh, the parallel algorithm/machine pair and its configuration. In order to support the solution of elliptic PDEs on fixed meshes, we are building a library of direct and iterative solution methods on the NCUBE, SEQUENT and VMC-SPAN machines and implementing a number of finite difference and finite element methods, [Hous 85], [Hous 87b], [Hous 88a], [Rice 84].

I.B Geometry Splitting Iterative Methods

The development of parallel algorithms for a given computation usually assumes the partitioning of the underlying computation into a number of subtasks with uniform computational load, minimum synchronization and minimum communication requirements. The various existing partitioning strategies for PDE computations can be divided into two groups; those that are based on some decomposition of the PDE domain or mesh and the ones that apply to the algebraic data structures of the discrete PDE system. We refer to the first group as *geometry decomposition* or *splitting* solvers and the second group as *matrix partitioning* solvers. In this project we are interested in PDE partitioning strategies based on some splitting of the discretized PDE domain, i.e., the geometry decomposition solvers. We are analyzing and implement various classes of iterative, capacitance and Schwarz type techniques for elliptic PDEs in two and three dimensional domains, [Care 88], [Clev 88], [Heum 87], [Mar 87]. Their usage is driven through a software geometry tool that generates automatic domain decompositions or allows their definition through an interactive mode.

II. PARALLEL PDE SOLVING SYSTEM

The goal of this project is the development of an expert system environment for solving PDEs on kMIPS, BIPS and SPAN machines. At the moment, it appears that there is a large gap between hardware and applications, [Gust 88], [Rice 87a], [Rice 87b], [Rice 87c], [Youn 88]. We are developing a software infrastructure that attempts to solve some hard problems introduced by the parallel processing of PDEs, [Gylus 76], [Hous 84], [Will 83].

II.A Overall Design Objectives

One of the main objectives of this system is the creation of a uniform programming environment in which parallel PDE solvers can be easily implemented and executed. In order to realize such a computational environment, we need to establish fixed data structures and interfaces among the various parts of the computation suitable for the implementation of parallel algorithms on kMIPS and BIPS architectures. Similar efforts have been used in the domain of sequential architectures with great success.

Parallel processing of an application requires the partitioning and allocation of the underlying computation to the targeted architecture. This is usually theoretically untractable problem. It is unrealistic to expect its solution by the novice user. This system provides an automatic and interactive facility that makes an attempt to hide the problem or gives the user the ability to solve it approximately by using high level software tools.

Very high level application oriented languages are becoming very popular means of man-machine interface. The proposed system will provide a PDE specification language for describing the problem and its solution requirements textually using notation similar to those in common engineering use. This interface will be combined with a geometry specification tool for defining the PDE domain and various operations to be performed on it.

We expect the system to be used for PDE modeling in production, educational and research environments. Thus it is necessary for the user to have control on various parameters of the solution process. In this system, the user will be able to control the mesh, geometry partition and mapping the computation to the targeted architecture.

Parallel processing provides many alternative solutions to a specific problem. Determining the efficiency of such solutions in a reliable way is a very crucial goal. This system will provide a uniform environment and testbed for obtaining performance measurements, [Bois 79], [Hous 87a], [Hous 87c], [Keye 87], [Rice 81], [Rice 82], [Nort 85].

The mesh/algorithm and configuration/machine selection problems are fundamental problems in parallel processing that must be solved. In general no analytic solution exists for these problems. We are studying an expert system heuristic solution to these problems and we are developing the appropriate infrastructure, [Chan 87], [Meir 85].

Finally, the visualization of the solution is very essential. This system will provide state of the art graphical displays of the solution and related data.

II.B Software infrastructure

The proposed infrastructure contains four major components, as shown in Figure 1; namely, a Man-Machine interface, an Expert System, a Partitioning/Allocation facility, and a library of PDE solving modules. The infrastructure is specified using an intermediate PDE problem specification language which allows direct access to the various tools of the system.

II.B.1 Man-machine interface

This user interface facility consists of a) dialogue tools, b) graphical facilities and c) geometry modeling and manipulation tools. The dialogue tools allow various dialogue styles such as "question and answer" mode, "command" mode, "text, expressions, statements, procedural language code" mode and various dialogue mechanisms such as keyboard, menu, forms and button input. The notations and terminology used are similar to those used in current engineering, mathematics and scientific work. The graphics facilities to be implemented in

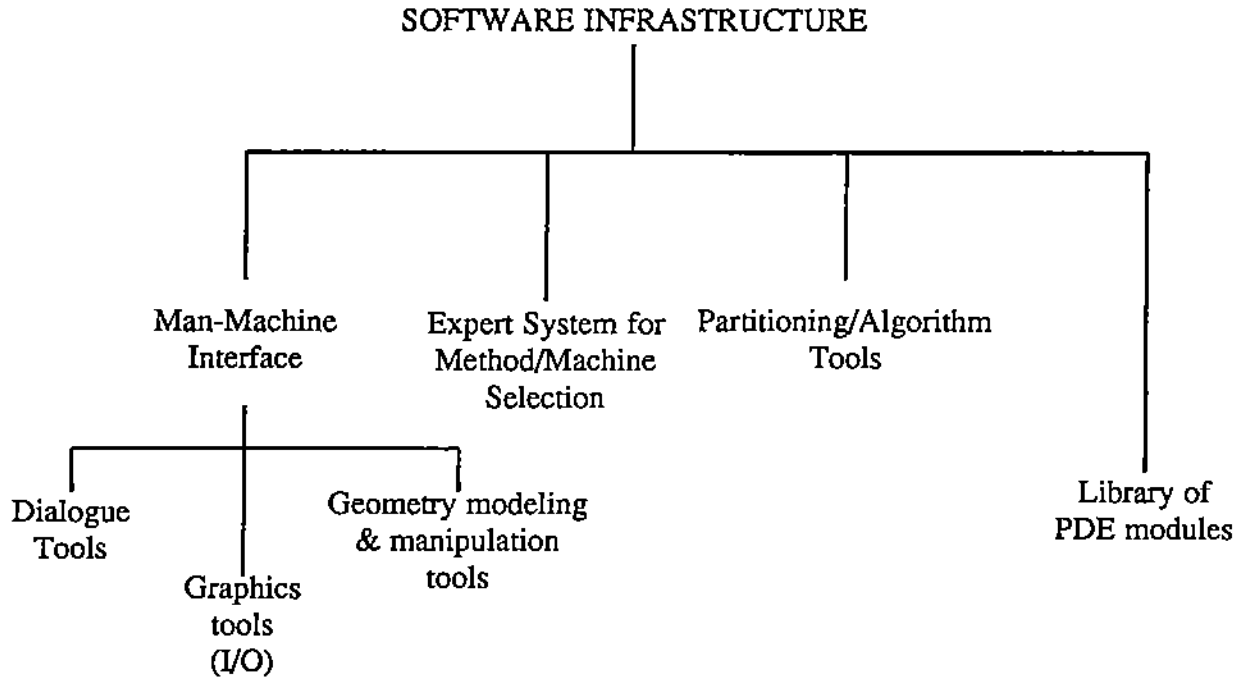

Figure 1. The //ELLPACK PDE software infrastructure. The four components are specified in an intermediate PDE oriented language.

conjunction with the PDE system output will provide both 2-D and 3-D plotting capabilities with scaling, rotation, translation, panning and zooming. The geometry tool will allow the definition of the PDE domain in an "expression" mode. It will provides geometry manipulation functions such as mesh definition using text and dialogue mechanism, domain partition (automatic/interactive) and editing of domain decompositions. This interface will be provided in a X11 windowing environment, allowing several windows to be active simultaneously and communicating with each other.

II.B.2 Expert system

This system assumes that the user specifies at least the *PDE problem*, *accuracy* and *time requirements* of the computation, together with its output specifications using the man-machine interface tools. The user has the option of explicitly specifying the solution modules, their corresponding parameters and the targeted architecture. In the absence of this information, the system makes the appropriate decisions based on existing knowledge available for the library modules. We are developing a knowledge based system for geometry splitting methods and a number of discretization schemes for parallel architectures. We are currently investigating the suitability and efficiency of various logic programming languages for scientific computing expert systems. We expect to experiment extensively with the new language SPINLOG of the SPAN project.

II.B.3 Intermediate PDE specification language

In the ELLPACK system [Rice 85], we have introduced a PDE language capable of specifying PDE equations, solution paths and their parameters, two dimensional PDE domains and output specifications. In the first phase of this project we will expand the ELLPACK language to accommodate parallel solution paths, domain decompositions, meshes, targeted machines and their characteristics. In the second phase we will extend the language to allow the definition of time dependent schemes and approximate operators. The language preprocessor will generate the control programs of the selected machine.

II.B.4 Partitioning and allocation tools for PDE computations

The software architecture of the PDE solving system is based on the assumption that an MIMD PDE solver consists of seven distinct tasks which communicate with each other through fixed interfaces. These tasks correspond to mathematical steps needed to obtain a PDE solution in a parallel MIMD computational environment. These steps are: a) PDE domain decomposition, b) PDE equation discretization, c) Preprocessing of the solution data structures, d) Partitioning of the underlying computation, e) Mapping of the underlying computation, f) Discrete equations solution, and g) Postprocessing of the output data. The synchronization of the above tasks depends on the way the partitioning and allocation of the PDE computation is implemented. It appears that there are two efficient methodologies for achieving these critical tasks. The first resolves this issue in terms of the PDE domain, while the other is applied to the precedence graph of the partitioned computation, [Chris 88a], [Chris 88b], [Hous 84], [Hous 88b], [Keye 87], [Taku 82], [Rodr 86], [Tang 87]. We have developed two software tools that implement these methodologies. They are referred to as the *Domain Decomposer* and the *Algorithm Mapper*. Figure 2 indicates the I/O requirements of these systems while Figures 3 and 4 show sample output of these tools. In the case of the domain decomposer, an interactive option, see Figure 5, exists where the user can specify or correct a domain partition. We plan to integrate these tools into the PDE solving system.

The Domain Decomposer currently operates on a mesh placed in a domain. The advantage of this approach is that simple, global data structures involving the unknowns of the PDE computation can be created at the beginning. The disadvantages are that creating these data structures for complex geometrics can be expensive and that a second independent data structure specifying the geometric relation of the subdomains must be created for the use of some algorithms. We are also exploring the alternative of decomposing the domain directly in geometric terms. This alternative requires considerable care in placing meshes in the subdomains so they match up efficiently. This approach gives more flexibility and, for some algorithms such as nested dissection, efficiency in solving the linear systems involved.

II.B.5 Architecture of the parallel PDE library

This topic is still an ongoing area of research, thus there are no established standards or well identified efficient data structures. We are experimenting with various alternatives. In our first implementation of this library we assume that the partition and mapping of the computation is determined on the discretization mesh using the geometry decomposer. The data of the domain decomposer are distributed in all processors. For the time being, in order to take advantage of the existing discretization modules at the front end or host of the parallel machine, we do global indexing of the unknowns and equations and broadcast it to individual processors. Each individual processor generates only the equations it is going to process which correspond to the various subdomains. The algebraic data are stored locally in sparse mode using the same data structures as ELLPACK with some additional structures for the interface or overlapping data. We currently study "block" and "substructuring" ordering schemes and the parallel software PDE architecture for geometry splitting methods in four phases. Phase 1 deals with domain discretization, partitioning and ordering of the underlying computation. This phase takes place in the front end or host of the parallel system. Phase 2 generates and stores the distributed data in the local memories of the available processors. Phase 3 carries out the solution of the discrete PDE system and Phase 4 deals with postprocessing of the solution and its derivatives. Notice that the last phase could take place asynchronously.

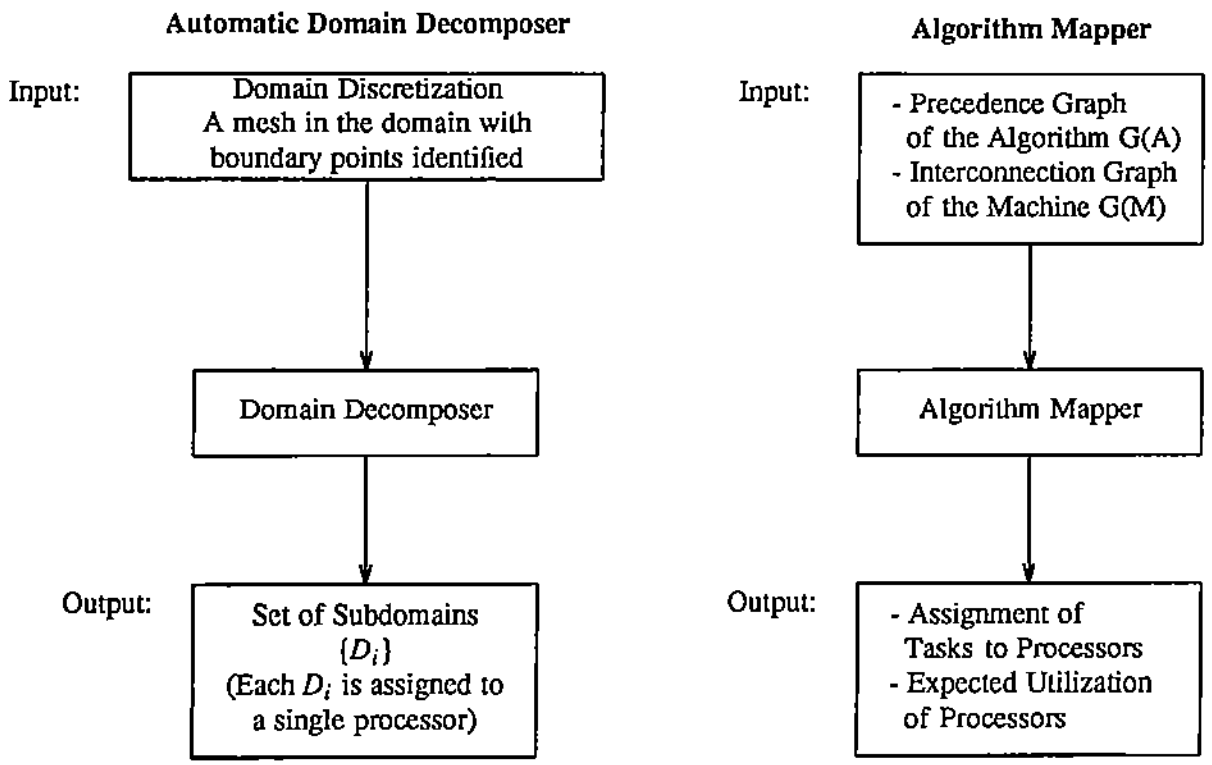

Figure 2. I/O requirements of the Domain Decomposer and Algorithm Mapper systems.

Figure 3. Display of a PDE domain, a finite element mesh and its domain decomposition obtained by the automatic decomposer for a parallel machine with 16 processors.

Figure 4. Input/Output of the algorithm mapper and its interface.

Figure 5. The user interface for the interactive geometry decomposer.

III. EXISTING HARDWARE INFRASTRUCTURE

The future computer hardware environment available to the users will be characterized by an array of computer hardware connected hierarchically through local area networks (LAN). We have access to a prototype future hardware facility consisting of X11 workstations connected through a LAN to four multiprocessor machines NCUBE (128 processors), ALLIANT FX/80(3 processors), SYMMETRY(12 processors) and FLEX/32(6 processors). The workstations support the user interface of the PDE system and preprocess some aspects of the computation. This includes the preprocessing of the language specifications, expert system support to man-machine interface, expert system solution to mesh/algorithm and configuration/machine selection problem, and partition/allocation of the underlying computation. The preprocessor of the user interface generates the control program(s) for the selected machine. For each of the machines we are developing a library of the PDE modules based on common data structures and fixed interfaces. The output will be displayed in the user workstation or in special devices connected to LAN or the machines. At present we are implementing the system on a SUN/NCUBE and SUN/VMC(SPAN) facilities.

IV. GEOMETRY DECOMPOSITION BASED METHODS FOR SOLVING PARTIAL DIFFERENTIAL EQUATIONS

The development of parallel algorithms for a given computation usually assumes the *partitioning* of the underlying computation into a number of subtasks with uniform computational load, minimum synchronization and minimum communication.

Recall from Section I.B that we are primarily interested in geometry decomposition solvers. These partitioning strategies usually depend on the characteristics of the meshes employed. The meshes used for spatial discretization can be characterized in at least three types: those that are *fixed* during the computation (*fixed meshes*), the ones that consist of a set of different *fixed* meshes (*multiple fixed meshes*) and those that vary during the computation (*variable meshes*). Our existing algorithmic/software infrastructure is applicable to computations defined on fixed meshes.

Domain discretization methods are normally defined in terms of *grid points* or *geometrical primitive elements*. Several partitioning techniques are based on various coloring schemes of grids or elements, [Adam 85], [Berm 84], [Olea 87]. These strategies very often lead to block algebraic data structures. In this project, we are exploring the potential of discretization methods which are formulated on some decomposition of the continuous or discrete PDE domain of definition in *subdomains* or *superelements* or *substructures*.

V. REFERENCES

[Adam 85] Adams, L., and Jordan, H., Is SOR Color-Blind? *SIAM J. Sci. Stat. Comput.*, 7, (1985), 490–506.

[Berm 84] Berman, F. and L. Snyder, On Mapping Parallel Algorithms in Parallel Architectures, *Proc. Internat. Conf. Parallel Processing*, (1984), 307–309.

[Bois 79] Boisvert, R.F., E.N. Houstis and J.R. Rice, A System for Performance Evaluation of Partial Differential Equations Software," *IEEE Trans.*

Software Eng., **19**, (1979), 418–425.

[Care 1988] Carey G., Kincaid D, Sepehrnoori K. and Young D., *Vector and Parallel Iterative Solutions of Large Sparse Linear Systems for PDEs*, Technical Report CNA-222, Center for Numerical Analysis, University of Texas at Austin, (1988).

[Chan 87] Chan T., Saad, Y., and Schultz, M., Solving Elliptic Partial Differential Equations on the Hypercube Multiprocessor, *Appl. Numer. Methods*, **3**, (1987), 81–88.

[Chris 88a] Christara, C.C., A. Hajidimos, E.N. Houstis and J.R. Rice, Geometry Decomposition Based Methods for Solving Elliptic PDEs, *Comp. Methods in Flow Analysis*, (H. Niki and M. Kawahara, eds.) Okouyama, Japan, pp. 175–182 (1988).

[Chris 88b] Christara, C.C., E.N. Houstis and J.R. Rice, A parallel Spline Collocation-Capacitance Method for Elliptic Partial Differential Equations, *Proceedings of the I.C.S. '88 Conference on Supercomputing*, (1988).

[Clev 88] Cleveland, B.W. and C.J. Ribbens, Schwarz Splitting Using ELLPACK, Virginia Polytechnic Institute, *TR-88-2*, May 1988.

[Gust 88] Gustafson, J.L., G.R. Montry and R.E. Benner, Development of Parallel Methods for a 1024-Processor Hypercube, *SIAM Journal of Scientific and Statistical Computing*, Vol. **9**, No. 4, July 1988.

[Gylus 76] Gylus, V., and D. Edwards, Optimal Partitioning of Workload for Distributed Systems, *Proc. Compcon*, (1976), 353–357.

[Hous 84] Houstis, C.E., E.N. Houstis and J.R. Rice, Partitioning and Allocation of PDE Computations In Distributed Systems, In PDE *Software: Modules Interfaces and Systems*, (1984). (B. Engquist, ed.) North-Holland, 67–85.

[Hous 85] Houstis, E.N., J.R. Mitchell and J.R. Rice, Collocation Software for Second-Order Elliptic Partial Differential Equations, *ACM Trans. Math. Software*, **11** (1985), pp. 379–412.

[Hous 88a] Houstis, E.N., J.R. Rice and E.A. Vavalis, Convergence of an $O(h^4)$ Cubic Spline-Collocation Method for Elliptic Partial Differential Equations, *SIAM J. Num. Anal.*, Vol. **25**, No. 1, 1988, 54–73.

[Hous 88b] Houstis, E.N., E.A. Vavalis and J.R. Rice, A Schwarz Variant of Cubic Spline Collocation for Elliptic PDEs, In *Hypercube Concurrent Computers and Applications*, G. Fox ed., ACM **4**, (1988), pp. 1746–1754.

[Hous 87a] Houstis, E.N., J.R. Rice, C.C. Christara and E.A. Vavalis, Performance of Scientific Software, In *Mathematical Aspects of Scientific Software*, J.R. Rice, ed., IMA **14**, (1987), pp. 125–154.

[Hous 87b] Houstis, E.N., M.A. Vavalis, and J.R. Rice, Parallelization of a New Class of Cubic Spline Collocation Methods, In *Advance in Computational Methods for Partial Differential Equations, VI*, (Stepleman and Vichnevetsky), IMACS, (1987), 167-174.

[Hous 87c] Houstis, C.E., E.N. Houstis, J.R. Rice and M. Samartzis, Benchmarking of Bus Multiprocessor Hardware on Large Scale Scientific Computing. In

Advances in Computational Methods for Partial Differential Equations, VI (Stepleman and Vichnevetsky), (1987).

[Keye 87] Keyes, D.E. and W.D. Gropp, A Comparison of Domain Decomposition Techniques for Elliptic Partial Differential Equations and Their Parallel Implementation, *SIAM J. Sci. Stat.*, Vol. **8**, No. 2, (1987), s166–s202.

[Mar 87] Marinescu, D.C. and J.R. Rice, (1987), Domain Oriented Analysis of PDE Splitting Algorithms, *J. Info. Sci*, **42**, to appear.

[McCa 87a] McCabe, S.C., PARLE – A Target Machine Language for Integrating Symbolic and Numeric Processing, SPAN Technical Report WP2-1 (1987).

[McCa 87b] McCabe, S.C., D.S. Djendov and A.N. Refenes, Virtual Machine Code Description, SPAN Technical Report WP1-3 (1987).

[Meir 85] Meier, U., A Parallel Partition Method for Solving Banded Systems of Linear Equations, *Parallel Comput.* **2,** (1985), 33–43.

[Neum 87] Neumann, M., and Plemmons, R., Convergence of Parallel Multisplitting Iterative Methods for *M*-Matrices, *Lin. Alg. Appl.*, **88,** (1987), 559–575.

[Nort 85] Norton, Alan and G.F. Pfister, A Methodology for Predicting Multiprocessor Performance, *Proc. Internat. Conf. Parallel Processing*, (1985), 772–778.

[Olea 87] O'Leary, D., Parallel Implementation of the Block Conjugate Gradient Algorithm, *Parallel Comput.* **5,** (1987), 127–140.

[Rice 81] Rice, J.R., E.N. Houstis, and W.R. Dyksen, A Population of Linear, Second Order, Elliptic Partial Differential Equations on Rectangular Domains, Parts 1 and 2 *Math. Comp*, **36**, (1981), 475–484.

[Rice 82] Rice J.R., Machine and Compiler Effects on the Performance of Elliptic PDE Software. In *Proc. IMACS 10th World Congress*, 1, IMACS, (1982), 446–448.

[Rice 84] Rice, J.R., Software Parts for Elliptic PDE Software. In *PDE Software: Modules, Interfaces and Systems*, (Engquist and Smedsaas. Eds), North-Holland, (1984), 123–134.

[Rice 85] Rice, J.R. and R.F. Boisvert, *Solving Elliptic Problems Using ELLPACK*, Springer-Verlag, New York (1985).

[Rice 87a] Rice, J.R., Supercomputing About Physical Objects. In *Proc. First Intl. Conf. Supercomputing*, Springer-Verlag, (1987).

[Rice 87b] Rice, J.R., ELLPACK: An Evolving Problem Solving Environment for PDEs. In *Problem Solving Environments for Scientific Computation* (B. Ford, ed), North-Holland, (1987), to appear.

[Rice 87c] Rice, J.R., Parallel Methods for PDEs. In *The Characteristics of Parallel Algorithms*, (Jamieson, Gannon and Donglass, eds.), Chapter 8, MIT Press (1987), pp. 209–321.

[Simo 87] Simon, C., SPINLOG Computational Model, SPAN Technical Report WP8-4 (1987).

[Taku 82] Takuhashi, Y., Partitioning and Allocations in Parallel Computations of Partial Differential Equations. In *Proc. 10th IMACS World Congress*, 1, IMACS, (1982), 311–313.

[Rodr 86] Rodrique, G., Some Ideas for Decomposing the Domain of Elliptic Partial Differential Equations in the Schwarz Process, *Commun. Appl. Numer. Method* **2**, (1986), 245–249.

[Tang 87] Tang, W.P., *Schwarz Splitting, A Model for Parallel Computations*, Ph.D. thesis, Stanford University, (1987).

[Will 83] Williams, E.A., Assigning Processes to Processors in Distributed Systems, *Proc. Internat. Conf. Parallel Processing*, (1983), 404–406.

[Youn 88] Young D., *The search for High Level Parallelism for the Solution of Large Sparse Systems*, Technical Report CNA-221, Center for Numerical Analysis, University of Texas at Austin, (1988).