2

REPORT DOCUMENTATION PAGE

Form Approved
OMB No.

**AD-A232 654**

REPORT TYPE AND DATES COVERED
Final Report, 1 Dec 89 to 30 Nov 90

NONLINEAR WAVES IN REAL MATERIALS

AFOSR-90-0075
61102F 2304/A3

James Glimm

Department of Applied Mathematics and Statistics
State University of New York
Stony Brook, NY 11794-3600

AFOSR-TR-

AFOSR/NM
Bldg 410
Bolling AFB DC 20332-6448

AFOSR-90-0075

Approved for public release;
distribution unlimited.

Using the interface methods developed by the principal investigator and co-workers, significant progress has been made in the studies of fluid flow focused on chaotic mixing and shock interactions. The studies were carried out only in two dimensions on sequential machines. In most of the cases for realistic problems of physical interest, further progress requires development of three dimensional algorithms. These algorithms, used for scientific purposes, including multiple computation to explore the solution dependence on physical parameters, will require extensive computer resources, which cost-effective parallel computations will allow. Our collective research supported by this grant has been to develop parallel algorithms for three dimensional fluid computations based on the interface methods. This research project is described in detail in the report.

| | | | |
|---|---|---|---|
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | SAR |

91 3 06 195

# FINAL REPORT

## Nonlinear Waves in Real Materials

Grant #: AFOSR 90-0075
Air Force Office of Scientific Research

James Glimm
Principal Investigator

Department of Applied Mathematics and Statistics
The University at Stony Brook
Stony Brook, NY 11794-3600

# Contents

# 1 SCIENTIFIC RESULTS

The advent of the computers thousands of times more powerful than those a decade ago totally altered the way scientists conduct research in the many areas for which an enormous amount of computer resources are required. A typical example is the study of nonlinear waves in real materials. Using the interface methods developed by the principal investigator and co-workers, we have made significant progress in the studies of fluid flow focused on chaotic mixing and shock interactions. The studies were carried out only in two dimensions on sequential machines. In most of these cases for realistic problems of physical interest, further progress requires development of three dimensional algorithms. These algorithms, used for scientific purposes, including multiple computation to explore the solution dependence on physical parameters, will require extensive computer resources, which cost-effective parallel computation will allow.

Our collective research supported by this grant has been to develop parallel algorithms for three dimensional fluid computations based on the interface methods. This particular research project formulated itself into five logical phases:

1. Parallelizing the 2D front tracking code and conducting performance analysis for optimization;
2. Developing 3D interface algorithms with parallel flavors;
3. Coding and tuning 3D algorithms in parallel;
4. Analyzing numerical results for gas dynamics to obtain a deeper understanding of hyperbolic systems;
5. Extending the parallel code to obtain solutions for elliptic systems.

To date when this report is written, phase I of the project is nearly completed with remaining research on parallel optimization including load balancing and communication issues. The three dimensional extensions, or the phase II, are also substantially advanced. We should report results from studies of phases III & IV in the near future, while phase V is still in its planning stage.

This report reviews in details our progress and short-term plans on all phases of the research supported by the Air Force Office of Scientific Research with additional support from other agencies.

## 1.1 Interface Methods: More Robust and Accurate

The finite difference method and many of its modified forms, with well-known simplicity in their implementations, will fail if they are (a) underresolved and (b) the problem being computed is sensitive to under resolution. We have identified areas where sensitivity to under resolution is important but was not realized. Our research, as summarized in the principal investigator's paper [5], has been to deal with this aspect of computational science, or to introduce more robust and accurate numerical methods.

A general theory has been developed [4], [ 6] to explain and predict sensitive behavior of nonlinear waves and wave interactions on internal length scales and for numerical computations on grid spacing and artificial viscosity. This theory develops a geometrical picture of wave interactions in terms of an intrinsically defined wave manifold. Bifurcation and length scale sensitivity are often associated with loss of transversality at wave curve crossings with admissibility boundaries and with saddle-saddle connections, as occurs for chemically reactive waves.

The interface methods were developed to response to the challenges that the variations of finite difference method face and are not able to overcome. In fact, the interface methods did overcome a series of obstacles. In particular the bifurcation of front topology, which results from colliding fronts, was solved in many cases. The methods produced remarkable accuracy for a wide range of scientific problems. As pointed out before, all these tests were performed in two dimensions. A major breakthrough will occur when realistic three dimensional problems of physical interest are solved. To solve a class of three dimensional problem efficiently, parallel computing will play a vital role. The research project focused around developing three-dimensional parallel interface methods has attracted most of our interest and energy.

Before investing in a full scale effort to developing three-dimensional parallel interface methods for a class of hyperbolic and elliptic systems, we have conducted an extensive study, reviewing the existing technology (hardware and software) to predict the feasibility of a project of this size. The major findings of the investigation are summarized in the following review article [5].

In this paper, we review the evolution of computer and computing technologies for the past 40 years, and we compare micro, mini, main-frame and supercomputers and justify the existence of shared-memory/distributed, MIMD/SIMD parallel computers. From these studies, we conclude that parallel computing is the wave of the future. There will be no scientific advancement in many areas without parallel computing. As shown by Table 1, it takes years of CPU time of the fastest supercomputers currently available, such as Connection Machine CM-2 with 64,000 processors, to solve some of large scale problems of current scientific importance.

## 1.2 Parallelization in Two Dimensions

Parallelizing our two-dimensional front-tracking code is the first step towards the goal of developing three-dimensional parallel interface methods.

We have successfully decomposed a two-dimensional material interface system into sub-interfaces (See Figure 1) and instructed the participating processors to build a self-contained sub-interface with artificial boundaries transferred from neighboring processors. The result is summarized in [3].

Conceptually, the technique of decomposing a computational domain into subdomains is rather standard.

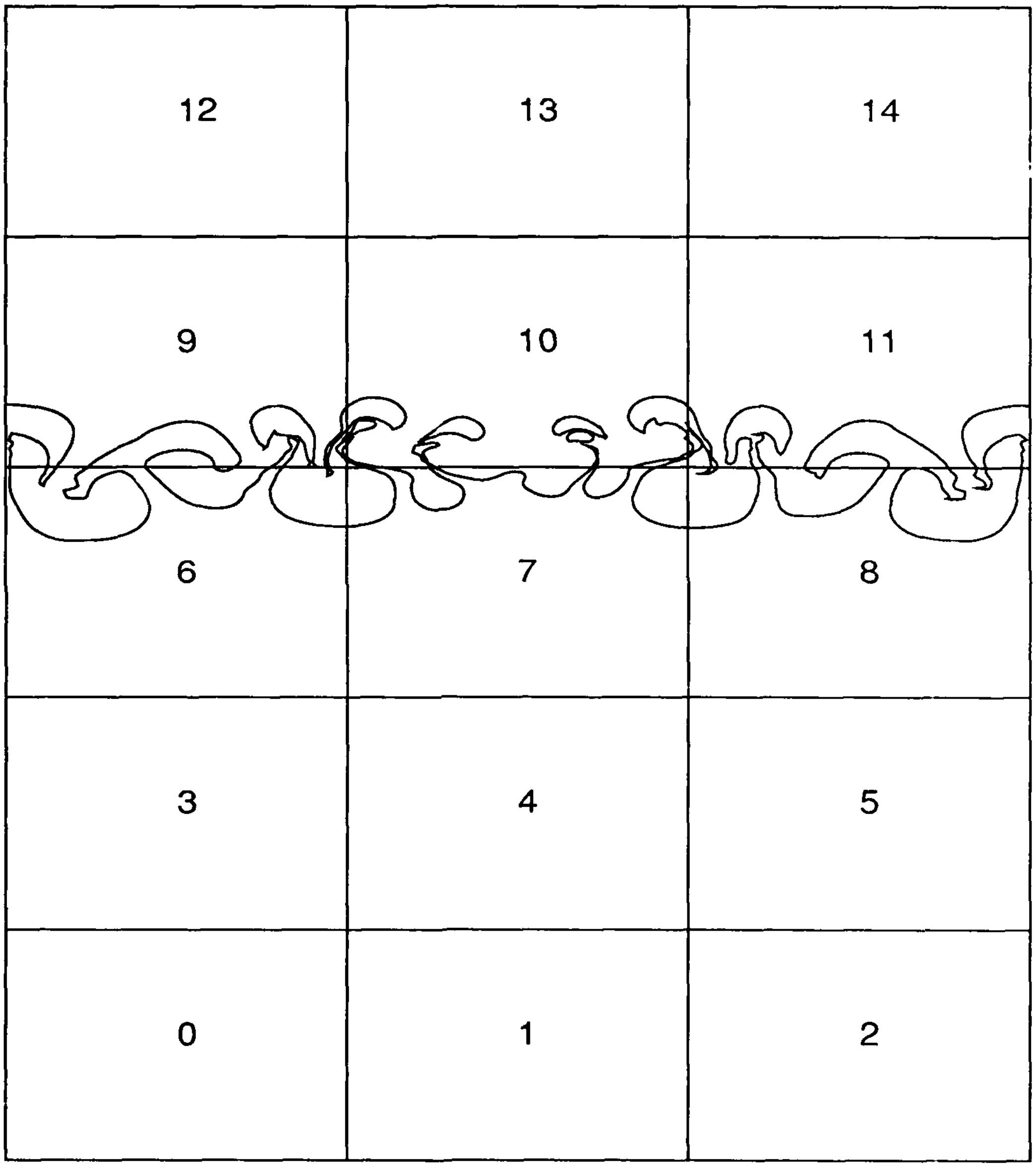

Figure 1: A material interface is decomposed into 15 subdomains (box-wise). The numbers in the sub-interfaces denote the processor ID's. Other possible decompositions (e.g. strip-wise) can also be carried out by the same code.

| Machines | Moderate Problems | Grand Challenge Problems |
|---|---|---|
| TFlop Machine | 2 seconds | 10 hours |
| CM-2 64K | 30 minutes | 1 year |
| CRAY Y-MP/8 | 4 hours | 10 years |
| Alliant FX/80 | 5 days | 250 years |
| SUN 4/60 | 1 month | 1.5K years |
| VAX 11/780 | 9 months | 14K years |
| IBM PC/8087 | 9 years | 170K years |
| Apple Mac | 23 years | 450K years |

Table 1: Typical times for solving medium size scientific problems and representative grand challenge problems on various computers.

First, we determine the border curves that bound the sub-interface and the find all curves that intersect these border curves. Then, we split these intersected curves into two pieces, one of which lies within the border, and is inserted into the sub-interface. Finally, we find the intersection points and insert them as boundary nodes into the sub-interface. The locally "interior" elements (nodes and curves) are inserted unchanged. This is part of the operations, namely the surgery on geometry of the objects. A more important but less complex operation is that of states assigned to these objects. We first set up the state parameters and then copy or compute via linear interpolation the states of points on a bond, or on curves or interior points.

From a higher level coding viewpoint, at initialization, individual processors are activated to generate correctly positioned sub-interfaces while at re-start from an existing full interface, individual processors are to pick up the right portion of the interface.

The decomposition scheme described above is also quite flexible, by generating reasonably arbitrary shaped sub-interfaces, in handling the load im-balance problems that appear in many calculations of physical interests. For example, in our simulation of interface systems, a heavier load appears mostly where the interface occurs. Moreover, a uniform distribution of the interface is not common. For a number of problems in the scientific applications, we are considering interfaces that frequently cluster in a horizontal (vertical) direction, which leads us to decompose the interfaces into vertical (horizontal) strip-wise sub-interfaces.

The process of constructing artificial boundaries is not as straightforward. First, all the processors need to figure out their position in the computational domain. One of the three cases can occur: (a) interior processor (with 8 surrounding processors in 2D and 26 in 3D), (b) boundary processor (with one and only one artificial boundary curve (surface in 3D) on the physical boundary curve (surface in 3D)), (c) corner processor (with one and only one subdomain corner on the physical subdomain corner). After figuring out its position, a processor begins to establish communication channels with a cluster of neighboring processors if possible. In any of these cases, a processor takes the necessary objects with state values from all its legal neighbors (eight in 2D if the processor is an interior processor, for example) or retains the original computational domain's boundary information to form its self-contained subdomain.

It is clear that propagating the updated self-contained sub-interface for an interface problem is like a serial code after artificial boundaries are set up correctly.

The subtlety in transferring the special data structures (objects) is unique to us. A more detailed description of these operations will appear in a separate publication.

Our tests are performed on two representative distributed-memory MIMD parallel supercomputers, the iPSC/860 and the nCUBE/2.

### 1.3 Three Dimensional Interface Methods

The main algorithmic issues for front tracking in three dimensions [3] are: (a) the construction of surface grids, (b) the construction of volume grids which are adapted to (i.e. which respect, or do not overlap with) specified surfaces, (c) the efficient computation of interface topology, (d) the resolution of self intersections in a tangled interface, and (e) parallel computation. Most of these issues are important for numerous methods of computation.

We have worked out the details of the first three parts, namely, the generation of the surface grid and unstructured interface fitting volume grid and the computation of interface topology. We are in the process of developing schemes to complete the last two parts.

In 3D, an interface consists of a set of surfaces. Each surface contains a set of triangles and is bounded by curves. A curve contains a set of doubly linked line segments called bonds. For a valid interface, there are no intersections between surfaces and no self-intersections on each surface. Surfaces may connect along boundary curves only. Each triangle has three pointers. They point to three neighboring triangles connected at its edges. If any edge of a triangle is on the boundary of the surface, the corresponding pointer points to the bond of a boundary curve which coincides with that boundary edge.

A timing study helped to resolve the major unknowns for predicting feasibility. For a major library (manipulating geometric objects) in three-dimensional interface methods, we have done extensive performance analysis on three typical machines: Intel iPSC/860, nCUBE/2 and a SUN SPARCstation-1 [2], [ 1]. The results indicate that significantly large three-dimensional physical problems can be solved with fairly high accuracy at a reasonable rate. We also estimate the relative performance of the three computers for our application. The SUN is two times faster than a one-processor nCUBE/2. The one-processor iPSC/860 is more than five times faster than one-processor nCUBE/2.

We can then project the best practical calculations we can perform.

What can we do on a single processor of the parallel machines? As we can see from Table 2, each time step that involves processing 50,000 triangles on a reasonable grid of

| Tris | Machine | N=20 | N=30 | N=40 | N=50 | N=80 | N=90 |
|---|---|---|---|---|---|---|---|
| 12288 | A | 12.2 | 10.5 | 10.0 | | | |
| | B | 33.7 | 26.1 | 24.6 | | | |
| | C | 52.0 | 44.2 | | | | |
| 24576 | A | 16.8 | 13.0 | | | | |
| | B | 44.5 | 33.6 | | | | |
| | C | 66.7 | 51.9 | | | | |
| 49152 | A | | | 10.1 | | | |
| | B | 60.9 | 40.1 | 32.1 | 29.1 | 25.0 | 25.0 |
| | C | 88.0 | 58.6 | 51.8 | 48.9 | | |
| 98304 | A | | | | | | |
| | B | 97.6 | 58.1 | 42.5 | 37.9 | 29.7 | 29.6 |
| | C | | | | | | |
| 196608 | A | | | | | | |
| | B | | 83.6 | 57.9 | 46.8 | 32.4 | 30.5 |
| | C | | | | | | |

Table 2: CPU time in units of minutes/mega-tri spent on three machines (A = single-processor iPSC/860, B = SUN and C = single-processor nCUBE/2) for grids $N \times N \times N$ with different numbers of triangles. The computational task associated with this test was to make the hash table for the triangles on the interfaces.

25 × 25 × 50 takes about 0.7 minutes on a single processor iPSC/860. Actually, we have just counted the time spent on the functions in the interface library. But, with some assumptions and the experience from the previous two dimensional calculations, we project that it takes $3-4$ times longer to complete a time step with all the calculations needed. Thus, we think a rate of 2.5 minutes per processor per time step is a proper estimate.

If we can gain a linear speedup with up to 64 processors, we can study problems on a moderate grid of 100 × 100 × 200 at a rate of 2.5 minutes per time step, which amounts to getting 500 time steps a day. It is the typical speed of most today's calculations.

## 2 PUBLICATIONS

[1] Y. Deng and J. Glimm, *Fluid Dynamics Using Interface Methods on Parallel Processors*, in Proceedings of the First International Conference on Computational Physics, Computational Physics Communication Supplement, North-Holland , 1990, to appear.

[2] ———, *Parallel Computations Using Interface Methods for Fluid Dynamics in Three Dimensions*, in Parallel CFD: Implementations and Results Using Parallel Computers, H. D. Simon, ed., The MIT Press , 1990, to appear.

[3] Y. Deng, J. Glimm, Y. Wang and Q. Zhang, *Parallel Algorithms for Fluid Interface Problems*, Transactions of the Eighth Army Conference on Applied Mathematics and Computing, 1990, to appear.

[4] F. Furtado, E. Isaacson, D. Marchesin and B. Plohr, *Stability of Solutions of Riemann Problems*, in preparation, 1990.

[5] J. Glimm, *Scientific Computing: von Neumann's Vision, Today's Realities and the Promise of the Future*, Proceedings of Symposia in Pure Mathematics, 50 (1990), p. 185.

[6] E. Isaacson, D. Marchesin, C. F. Palmeira and B. Plohr, *The Global Structure of Nonlinear Waves in Conservation Laws*, in preparation, 1990.

# 3 SUPPORTED PERSONNEL

## 3.1 Senior Personnel

1. Qiang Zhang

2. James Glimm

3. Brad Plohr

## 3.2 Postdoctoral Fellows

1. Yuefan Deng

2. Yi Wang

## 3.3 Graduate Research Assistants

1. Yupin Chen

# 4 APPENDIX: Abstracts of Publications

# Parallel Computations Using Interface Methods for Fluid Dynamics in Three Dimensions

Yuefan Deng[*]
James Glimm[†]

*Department of Applied Mathematics and Statistics*
*SUNY at Stony Brook, Stony Brook, NY 11794-3600*

## Abstract

The value of interface methods or front tracking has been demonstrated by a series of computations for two dimensional fluid flow focused on chaotic mixing, shock interactions and oil reservoir simulation. In most of these cases, further progress depends upon three dimensional computations. These computations, used for scientific purposes, including multiple computation to explore the solution dependence on physical parameters, will require extensive computer resources. Our preliminary timing study indicates that a complete and reliable calculation in three dimensions on a reasonably fine grid will take months on a sequential machine with a peak performance of 15 MIPS. However, the newly introduced Intel iPSC/860 parallel supercomputer, even in the smallest configuration with 8 processors and a total peak rate of 480 Mflops, will reduce the time to days, based on very conservative estimates of peak utilization.

The complete simulation code exceeds one quarter of a million lines of C code, including support for multiple physical applications. The

[*] Supported by the Applied Mathematics Subprogram of the U.S. Department of Energy DE-FG02-90ER25084, the Army Reseach Organization, grant DAAL03-K-0017.

[†] Supported also by the National Science Foundation, grant DMS-89018844.

MIMD hypercube architecture of the Intel machine allows a feasible route to parallelization of this code, especially for explicit time step algorithms. In fact, each processor will run identical instructions, which are nearly identic_l to the sequential algorithms, but restricted to a subdomain, with communication of boundary information once or a limited number of times per time step. The advantage of relatively small communication cost compared to computation thus provides an ideal environment for domain decomposition.

The main algorithmic issues for front tracking in three dimensions are presented and analyzed. Based on the idea of domain decomposition and the results of timing studies, we present arguments supporting the feasibility and justifying the significance of introducing parallel computing into this investigation.

# 1 Fluid Dynamics Using Interface Methods on Parallel Processors

Yuefan Deng and James Glimm [1]

Abstract. A general idea is introduced to develop parallel algorithms for three dimensional fluid computations based on interface methods. These methods offer unique resolution capabilities which have been demonstrated by a series of sequential computations for two dimensional fluid flow focused on chaotic mixing, shock interactions and oil reservoir simulation. In most of these cases, further progress requires development of three dimensional algorithms. These algorithms, used for scientific purposes, including multiple computation to explore the solution dependence on physical parameters, will require extensive computer resources, which cost-effective parallel computation will allow.

A complete simulation code exceeds one quarter million lines of C code. The MIMD architecture of the Intel machine and the interface algorithm's inherent characteristics allow a feasible parallelization in terms of space and time. Locality minimizes boundary information exchange and the explicit time step provides convenient synchronization.

Our preliminary timing study indicates that a complete and reliable calculation in three dimensions on a reasonably fine grid will take over a month on a typical workstation with a peak performance of 15 MIPS/ 2 Mflops. However, the same amount of work takes Intel's iPSC/860 parallel supercomputer, even in a small configuration with 16 processors and a total peak rate of near 1 GIPS / 1 Gflop, about a day.

The main parallel algorithmic issues for front tracking in three dimensions are presented and analyzed. Based on these parallelization ideas and the results of timing studies, we present arguments supporting the feasibility and justifying the significance of introducing parallel computing into this investigation

[1] Both authors are in the Department of Applied Mathematics and Statistics, SUNY at Stony Brook, Stony Brook, NY 11794-3600

Proceedings of Symposia in Pure Mathematics
Volume 50 (1990)

# Scientific Computing: Von Neumann's Vision, Today's Realities, and the Promise of the Future

JAMES GLIMM

**1. Von Neumann's vision.** Science was transformed by the invention of calculus. The impact of computers upon science will be at least as large.

Across a broad front, computers will allow the transition from qualitative to quantitative and from descriptive to predictive. John von Neumann foresaw that the equations describing scientific phenomena, once expressed in mathematical terms, could be solved numerically, without recourse to routine or repetitive experiment.

This vision is nothing less than the second half of the scientific revolution. Throughout four centuries we have expected that a successful scientific theory would have its major concepts expressed quantitatively as numbers and its major relationships expressed as mathematical equations: the truth of this theory was settled by experimental tests and hand calculations, often in idealized situations. The second half of the scientific revolution is no less sweeping in its goals. The solutions of the equations are also to be obtained on mathematical grounds, by numerical computation, without restriction to idealized cases.

Von Neumann is the founding father of modern scientific computing. His work in the major areas of this field—numerical analysis, numerical algorithms, computations, mathematical modeling, and asymptotic analysis—stands today as vital and seminal. In this lecture, we will trace several of

1980 *Mathematics Subject Classification* (1985 *Revision*). Primary 76N10, 65M05, 35L65.

*Key words and phrases.* conservation laws, artificial viscosity, shock waves, chaos.

Supported in part by the National Science Foundation, grant DMS-8619856; the Applied Mathematical Sciences subprogram of the Office of Energy Research, U. S. Department of Energy, under contract DE-FG02-88ER25053; the Army Research Office, grant DAAG29-85-0188; and the Air Force Office of Scientific Research AFOSR-88-0025.

This paper is in final form and no version of it will be submitted for publication elsewhere.

# THE GLOBAL STRUCTURE OF NONLINEAR WAVES IN CONSERVATION LAWS I: FRAMEWORK

ELI L. ISAACSON
DAN MARCHESIN
C. FREDERICO PALMEIRA
BRADLEY J. PLOHR

January, 1991

ABSTRACT. We introduce a unifying framework for treating all of the fundamental waves occurring in general systems of $n$ conservation laws. After trivial solutions have been eliminated by means of a blow-up procedure, pairs of states satisfying the Rankine-Hugoniot condition form an $(n+1)$-dimensional manifold $\mathcal{W}$. There is a distinguished $n$-dimensional submanifold of $\mathcal{W}$ containing a single one-dimensional foliation that represents the rarefaction curves for all families. Similarly, there is a foliation of $\mathcal{W}$ itself that represents shock curves. We identify other $n$-dimensional submanifolds of $\mathcal{W}$ that are naturally interpreted as boundaries of regions of admissible shock waves. These submanifolds also have one-dimensional foliations, which represent curves of composite waves.

This geometric framework promises to simplify greatly the study of the stabiliy and bifurcation properties of global solutions of Riemann problems for mixed elliptic/hyperbolic systems. In particular, bifurcations of wave curves can be understood as resulting from loss of transversality between foliations and admissibility boundaries.

1980 *Mathematics Subject Classification* (1985 *Revision*). 35B32, 35L65, 35L67, 35L80, 57R45.

***Key words and phrases.*** conservation laws, Riemann problems, wave curves, bifurcation.

This work was supported in part by: a grant from the NSF/CNPq U. S.-Latin America Cooperative Science Program; the Institute for Mathematics and its Applications with funds provided by the National Science Foundation; the Air Force Office of Scientific Research under Grant AFOSR 90-0075; the U. S. Army Research Office under Grant DAAL03-87-K-0028; the Financiadora de Estudos e Projetos; the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq); the Fundacão de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ); the Coordenacão de Aperfeiçoamento de Pessoal de Ensino Superior (CAPES); and the Sociedade Brasileira de Matemática (SBM) .

Typeset by $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX

# PERSPECTIVES ON PARALLEL COMPUTING

Yuefan Deng and James Glimm [*]
Department of Applied Mathematics and Statistics
SUNY at Stony Brook, Stony Brook, NY 11794-3600

David H. Sharp[†]
Theoretical Division
Los Alamos National Laboratory
Los Alamos, NM 87545

## 1 Why Compute Faster?

Technology, information and knowledge are critical commodities, whose use is tied to our ability to compute. The computer is a critical technology for the competitive position of major industries. It is critical for our national security and increasingly, it appears as an aspect of medical and biological understanding. The following table (Table I.1) illustrates the times for solving representative medium sized and grand challenge scientific problems on typical computers, including one "imaginary Teraflop machine" due by 1995. The basis for comparison is an estimate of sustained performance. These numbers in the table are accumulated by experience, and for the newer computers are not yet determined precisely. For more established results, we obtain the numbers from published sources, based on careful benchmarks and studies (Refs. [1] and [2].)

We want to compute better in order to understand more. Total computational performance is measured by system and by human criteria. System performance is measured in terms of speed, memory and I/O (input and output) capabilities. The human criteria are the knowledge of computational algorithms, or ways of translating ideas and formulas into computer language, and the problem based understanding, or science, put into the problem. Parallel computing is a method to improve system performance dramatically. We examine the significance of this very profound development.

### 1.1 Society and Science in the Computer Age

Just as machinery has lifted the deadening weight of manual labor from our backs, the computer will remove the weight of repetitive mental tasks from our minds. The tasks involve not only words and numbers, but the control of the machines themselves, and extend to the organization of society. Thus the computer completes the industrial revolution, begun with the invention of the steam engine, and ushers in its own era.

[*] Supported by the Applied Mathematics Subprogram of the U.S. Department of Energy DE-FG02-90ER25084, the National Science Foundation, grant DMS-89018844, and the AFSOR, grant 90-0075.

[†] Supported by the U.S. Department of Energy.

# PARALLEL ALGORITHMS FOR FLUID INTERFACE PROBLEMS [*]

Yuefan Deng, James Glimm,
Yi Wang and Qiang Zhang

Department of Applied Mathematics and Statistics
The University at Stony Brook
Stony Brook, NY 11794-3600

## Abstract

In this report, we present and analyze the results of applying parallel algorithms to a two dimensional gas dynamics code using front tracking. We also discuss the ideas to generalize the algorithm to three dimensions. The main purpose of this paper is to demonstrate that parallel computations can be applied to complex algorithms. We take front tracking as an example.

A number of computations for two dimensional fluid flow have been successful to simulate chaotic mixing, shock interactions and oil reservoir simulation. In most of these cases, further progress depends upon three dimensional computations. Performing systematic studies of chaotic mixing in three dimensions will require the extensive computer resources which parallel computations can offer.

In order to develop a reliable parallel programming paradigm for three dimensional studies, we first parallelize the relatively simpler serial two dimensional gas dynamics code on two representative distributed-memory MIMD parallel supercomputers, the iPSC/860 and the NCUBE/2.

The main algorithmic issues for front tracking in three dimensions are: (1) the construction of surface grids, (2) the construction of volume grids which are adapted to (i.e. which respect, or do not overlap with) specified surfaces, (3) the efficient computation of interface topology, (4) the resolution of self intersections in a tangled interface, and (5) parallel computation. Most of these issues are important for numerous methods of computation. We present methods for addressing these issues which are appropriate to the front tracking context.

[*] Supported by the Applied Mathematics Subprogram of the U.S. Department of Energy DE-FG02-90ER25084, the National Science Foundation, grant DMS-9001884, and the AFSOR, grant 90-0075.