AIIM

Association for Information and Image Management

1100 Wayne Avenue, Suite 1100
Silver Spring, Maryland 20910
301/587-8202

Centimeter

Inches

MANUFACTURED TO AIIM STANDARDS
BY APPLIED IMAGE, INC.

O

Three-Dimensional Gyrokinetic Particle-In-Cell Simulation of Plasmas on a Massively Parallel Computer: *Final Report on LDRD Core Competency Project* *FY91—→FY93*

Jack A. Byers, Principal Investigator[†]
Timothy J. Williams, Co-Investigator
Bruce I. Cohen, Co-Investigator
Andris M. Dimits, Co-Investigator

MFE Theory and Computations
Lawrence Livermore National Laboratory

April 27, 1994

1 Overview

1.1 Purpose of Project

One of the programs of the Magnetic Fusion Energy (MFE) Theory and Computations Program is studying the anomalous transport of thermal energy across the field lines in the core of a tokamak. We use the method of gyrokinetic particle-in-cell simulation in this study. For this LDRD project we employed massively parallel processing, new algorithms, and new formal techniques to improve this research. Specifically, we sought to take steps toward: reaching experimentally-relevant parameters in our simulations, learning parallel computing to have as a resource for our group, and achieving a 100× speedup over our starting-point Cray2 simulation code's performance.

1.2 Accomplishments

We have learned much about the art of parallel processing. We ported our three-dimensional simulation code to various parallel processors using three different parallel programming paradigms.

We have made significant advances in using new algorithms and formal techniques to enhance our gyrokinetic simulations. We devised and published a semi-implicit orbit-averaging method. We implemented the new partially-linearized δf method to greatly reduce computational noise. We have used quasiballooning coordinates to more efficiently model wave structures with a given grid size.

Using these more efficient algorithms, we have drawn closer to tokamak experimental parameters in our gyrokinetic core turbulence simulations. This year the four authors on this project will present simulation results at the IAEA fusion meeting.

[†]The original PI, Yoshi Matsuda, died in 1991.

From our starting-point simulation code, which ran at 120 microseconds per particle per timestep on the Cray2 in 1991, we have achieved a performance improvement of $100\times$. Our generic parallel δf code, running on 512 nodes of the CM5, runs at 1 microsecond per particle per timestep on a problem whose parameters are taken from our current production suite.

Finally, our work on this project has helped secure a place in the Numerical Tokamak Project, an HPC Grand Challenge. This project now brings new funding to our group.

2 Details

2.1 Scientific Motivation

One of our group’s core programs is the study of low-frequency microinstabilities in tokamak plasmas, which are believed to cause energy loss via turbulent thermal transport across the magnetic field lines. This “anomalous” transport of energy radially out of the plasma is partly responsible for overall loss of energy confinement in the devices. Understanding this turbulent transport is a fundamental outstanding question in MFE theory, and a question of great interest to the experimentalists. [16] This question has inspired, among other efforts, the DOE Transport Task Force Initiative.

2.2 Computational Challenge

An important tool in this study is gyrokinetic particle-in-cell (GPIC) simulation. [17] Gyrokinetic, as opposed to fully kinetic, methods are particularly well suited to the task because they are optimized to study the frequency and wavelength domain of the microinstabilities. Furthermore, many researchers now employ low-noise δf methods [10] [18] to greatly reduce statistical noise by modeling only the perturbation of the gyrokinetic distribution function from a fixed background, not the entire distribution function.

In spite of the increased efficiency of these improved algorithms over conventional PIC algorithms, GPIC simulations of tokamak microturbulence are still highly demanding of computer power—even fully-vectorized and multiprocessing codes on vector supercomputers. For this reason, we have worked for several years to redevelop these codes on massively parallel computers. During the last year of this LDRD project, this computational challenge has been formalized as part of a funded HPC Grand Challenge project: the Numerical Tokamak Project.

2.3 Review of Our Efforts

2.3.1 Parallel Computation

As part of this LDRD research, we investigated three distinct parallel programming paradigms as applied to our starting-point three-dimensional (3D) GPIC code: data-parallel, control-parallel, and message-passing. This has not only given us exposure to a broad spectrum of parallel processing techniques, but has led us to develop what we believe is the most portable and efficient parallel code for today’s machines. (The reader is referred to earlier publications for definitions and elaborations on these paradigms not contained in this report. Specifically, Refs. [11] and [3] describe the inefficiencies we found with the first two approaches, and how they motivated efforts on the third.)

First, we developed a data-parallel code using CM Fortran on the Thinking Machines CM2. This paradigm uses Fortran-90 array syntax to express parallel operations over many data-array elements; the compiler and runtime system handles distribution of global data arrays among the memories of the processors. We found that the random, dynamically-changing communication

patterns between grid and particle data arrays became the chief component of execution time in this code, and performance on the CM2 relative to the Cray was limited by this. [11] [3]

Second, we developed a control-parallel version using the Parallel Fortran Preprocessor. [15] This paradigm allows both shared and local data structures. Special syntax provides for nested parallel loops and processor-team splitting to statically allocate work (like loop iterates) among the processors. We stored the grid arrays in the PIC algorithm as shared arrays, using the shared interleaved memory system on the BBN TC2000; all other data structures were local. We experimented with various approaches to handling grid-particle data communication, all of which ultimately limited parallel efficiency as more processors were employed on a fixed problem size. [11] [3]

Third, after studying inefficiencies in the first two paradigms, we developed a generic parallel code based on domain decomposition and message passing. In this paradigm, the computational domain of the simulation is decomposed among the processors, and each runs the standard simulation algorithm on the data in its own subdomain (particle and grid). Communication to handle data at the borders of the subdomains is done explicitly by bundling and sending data as interprocessor messages. In this last approach, we have found the most satisfying results—a code that is easily portable to many different machines and which has predictable performance characteristics. We have demonstrated this code on clusters of workstations, the CM5, the TC2000, and the C90. Using fundamental ingredients of floating-point operation speeds, data communication bandwidths, and message-startup latencies, we have derived a comprehensive performance model to describe a complete timestep of our GPIC simulation. We have found excellent agreement between predicted and measured behavior. Furthermore, we have predicted scaling with good parallel efficiency to thousands of processors. Figure [1] shows both that excellent agreement between model and measured performance and scaling with high efficiency up to 4096 processors for a problem size in our current suite of production runs. For the larger runs we anticipate as part of the Numerical Tokamak Project, we could effectively use $O(10000)$ processors if they were available. It is interesting to note that these measurements were made on the CM5 using only the SPARC chips for floating-point arithmetic (TMC does not provide vectorizing compilers to access their $10\times$ faster vector units). We predict much better performance on machines like the Cray T3D, which will offer both much faster floating-point performance and higher network bandwidth; new machines like this one should run our simulations ten times faster than a 16-processor C90.

2.3.2 Algorithm and Method Development

We have investigated several new algorithms and formal techniques to improve the intrinsic efficiency in all our GPIC codes. These included semi-implicit orbit-averaging algorithms, partially-linearized δf algorithms, and quasiballooning coordinates.

We formulated [12] and implemented [9] a semi-implicit orbit-averaged algorithm as a way to exploit multiple timescales to improve efficiency. Formally, this approach removes some stability constraints on the timestep by using implicit timestepping. In practice, we found marginal gains, since for the physics parameters of our microinstability studies the nonlinear velocities overwhelmed the velocities used in formulating and analyzing the new algorithm. This work has proved invaluable, however, in more recent formulations of moment-implicit δf algorithms which will allow us to efficiently include more realistic electron physics in all our work. [4] [2]

Like many others in our field, we have switched to using the partially-linearized δf method to increase efficiency. In this method, the particles sample the perturbation of the phase-space distribution function away from a fixed background rather than the entire distribution function. This, when coupled with a partial linearization of the equations, can result in greatly reduced simulation noise—allowing the use of fewer particles. [10] [18] We implemented this method in all

Figure 1: Performance model and measured performance on a production-sized run [$256 \times 128 \times 32$ grid, 8M particles (8 per cell)]: (a) Absolute timing in μsec per particle per timestep. (b) Parallel efficiency, defined as $\frac{1}{N_{\text{procs}}} T_1 / T_{N_{\text{procs}}}$; T_1 is the time for an optimal serial algorithm on 1 processor, which our code uses. This problem was much too large for 1 CM5 node, hence no parallel efficiency measurements; measurements on smaller problems agree well with the model predictions.

our GPIC codes, for all platforms.

In order to more efficiently represent the structures in the simulated instabilities with a given grid resolution, we have begun to use a modified coordinate system called quasiballooning coordinates. [5] We now have a 3D code with correct toroidal geometry and boundary conditions to model a flux tube (section of a toroidal annulus that approximately follows the magnetic field lines). Production simulations underway in the SPP queue on the NERSC C90 represent the first studies of toroidal ion temperature gradient instabilities in the presence of externally imposed sheared toroidal flows—a unique capability of our production code. We are currently implementing the quasiballooning-coordinate algorithm under the parallel GPIC framework developed for the slab simulation code.

2.3.3 Tool Building

In the process of building the generic message-passing 3D slab GPIC code, we have developed several useful tools for parallel programs. For example, we have developed a routine that transposes a 3D data array with an arbitrary uniform 3D domain decomposition and number of guard cell layers to an arbitrary different 3D domain decomposition. This plays a central role in a generic parallel 3D real-to-complex FFT routine which we also developed. We have also pioneered the use of a public-domain portable binary I/O library, NetCDF, for doing asynchronous parallel I/O on multiple platforms. The output files from our simulations, written using this system, are independent of the machine architecture and the processor count or domain decomposition. In conjunction with the DDI software developed at NERSC, which reads this file format, we use graphics workstations to postprocess our data with commercial visualization software. To make our code portable across multiple hardware and software systems, we have hidden the system-specific message-passing details within a library of generic wrapper functions. So far, we have wrapped several message-passing libraries: PVM (for workstations and the C90), CMMD (for the CM5), and LMPS (for the TC2000).

These tools have already seen some reuse by other computational scientists. Another participant in the Numerical Tokamak Project has taken the parallel FFT, with its domain-decomposition transposer, to use in an object-oriented parallel PIC system. The Earth System Modeling group at LLNL has adopted the NetCDF parallel I/O techniques. Researchers at PPRI at LLNL are benefiting from the message-passing wrapper library in developing a parallel dynamic alternating direction implicit (DADI) algorithm.

2.4 Benefits

We have gained a solid position in the world of parallel scientific computing. We are ready with simulation codes to run on the proposed NERSC MPP machine. Our new skills are attractive to other organizations within and outside the lab. We are collaborating with PPRI on a parallel DADI solver. We are establishing collaboration on seismic-wave propagation simulations on MPP's. T. Williams is now supported by the NERSC parallel computing group to help other scientific programmers move applications to MPP's. Finally, and most importantly, this work was instrumental in our getting a place in the HPCC Numerical Tokamak Project, which is a new source of funding for our group.

References and Publication List

Our Publications

Following is a partial list of publications that derive, at least in part, from the research described in this report.

1. A. M. Dimits and B. I. Cohen, "Collision Operators for Partially Linearized Particle Simulation Codes," *Phys. Rev. E* **49**, 709 (1994).
2. B. I. Cohen *et al.*, "Implementation of a δf Implicit Moment Algorithms," *Proc. International Sherwood Fusion Theory Conference*, March 14-16, Dallas (1994).
3. Timothy J. Williams, "3D Gyrokinetic Particle-In-Cell Simulation of Fusion Plasma Microturbulence on Parallel Computers," *Proc. 1993 SCS Simulation Multiconference High Performance Computing Symposium*, Arlington, VA, March 29-April 1, 114-119 (1993).

4. B. I. Cohen *et. al.*, "Formulation of δf Implicit Moment Algorithms for Particle Simulation of Turbulence," *Bull. Am. Phys. Soc.* **38** 2016 (1993).
5. A. M. Dimits, "Fluid Simulations of Tokamak Turbulence in Quasiballooning Coordinates," *Phys. Rev. E* **48**, 4070 (1993).
6. B. I. Cohen, Timothy J. Williams, Andris M. Dimits, and Jack A. Byers, "Gyrokinetic simulations of $\mathbf{E} \times \mathbf{B}$ velocity-shear effects on ion-temperature-gradient modes," *Phys. Fluids B* **5**, 2967 (1993).
7. Nathan Mattor, Timothy J. Williams, and Dennis W. Hewett, "Algorithm for Solving Banded Diagonal Matrix Problems in Parallel," UCRL-JC-114756 (1993) (Submitted to *Parallel Computing*).
8. A. M. Dimits and B. I. Cohen, "Simulation Models for Tokamak Plasmas," *Proc. 14th IAEA Conference on Advances in Simulation and Modeling of Thermonuclear Plasmas*, IAEA Technical Meeting, Montreal, Canada June 15-17, 1992 (Vienna, 1993).
9. B. I. Cohen and Timothy J. Williams, "Implementation of a Semi-Implicit Orbit-Averaged Gyrokinetic Particle Code," *J. Comput. Phys.* **107**, 282 (1993).
10. A. M. Dimits and W. W. Lee, "Partially Linearized Algorithm in Gyrokinetic Particle Simulation," *J. Comput. Physics* **107**, 309 (1993).
11. Timothy J. Williams and Y. Matsuda, "3D Gyrokinetic Particle-In-Cell Codes On The TC2000 And CM2." In *The 1992 MPCJ Yearly Report: Harnessing the Killer Micros*, E. R. Brooks *et. al.* eds., UCRL-ID-107022-92,303-311 (1992).
12. B. I. Cohen and Timothy J. Williams, "Semi-Implicit Particle Simulation of Kinetic Plasma Phenomena," *J. Comput. Phys.* **97**, 224 (1991).
13. A. M. Dimits, "Efficient Simulation of High- n Turbulence in Ballooning Coordinate," *Proc. International Sherwood Fusion Theory Conference*, April 22-24, Seattle (1991).
14. Timothy J. Williams, Y. Matsuda, and E. Boerner, "Parallel Particle Simulation On the TC2000 and CM-2," In *The 1991 MPCJ Yearly Report: Attack of the Killer Micros*, E. R. Brooks *et. al.* eds., UCRL-ID-107022, 133-138.

Other References

Work by other authors cited in this report.

15. B. C. Gorda and K. H. Warren, "PFP: A Scalable Parallel Programming Model," in *The 1992 MPCJ Yearly Report: Harnessing the Killer Micros*, E. R. Brooks *et. al.* eds. Lawrence Livermore National Laboratory publication UCRL-ID-107022-92, 17-21 (1992).
16. J. D. Callen, *Phys. Fluids B* **4**, 2142 (1992).
17. W. W. Lee, *J. Comput Physics* **72**, 243 (1987).
18. M. Kotschenreuther, "Particle Simulations with Greatly Reduced Noise," *Proc. 14th International Conference on Numerical Simulation of Plasmas*, Annapolis, MD (1987).

**DATE
FILMED**

6 129/94

END