PCCM2: A GCM ADAPTED FOR SCALABLE PARALLEL COMPUTERS

John Drake*, Ian Foster+, James J. Hack++, John Michalakes+,
B. David Semeraro*, Brian Toonen+, David L. Williamson++,
Patrick Worley*

+ Argonne National Laboratory
* Oak Ridge National Laboratory
++ National Center for Atmospheric Research

1. INTRODUCTION

The Computer Hardware, Advanced Mathematics, and Model Physics (CHAMMP) program (Department of Energy 1990) seeks to provide climate researchers with an advanced modeling capability for the study of global change issues. Current general circulation models are usually applied at coarse spatial resolution with minimal coupling between ocean, atmosphere, and biosphere. As a first goal in the program, state-of-the-art models are being adapted for execution on scalable parallel computers. (A parallel computer is said to be *scalable* if its performance can be increased by adding processors. The Intel Paragon, IBM SP1, Thinking Machines Corporation CM5, and Cray T3D are examples: each can incorporate $10 - 10^3$ microprocessors.) Accomplishment of this task will lay the groundwork for the coupling of oceanic and atmospheric models to produce advanced climate models with improved process representations and capable of exploiting the teraflops computers that are expected to become available later this decade.

One of the more ambitious projects being undertaken in the CHAMMP program is the development of PCCM2, an adaption of the Community Climate Model (CCM2) for scalable parallel computers. This project involves a team of computer scientists, applied mathematicians, and climate modelers at Argonne National Laboratory (ANL), Oak Ridge National Laboratory (ORNL), and the National Center for Atmospheric Research (NCAR). In September 1993, a significant milestone was reached when PCCM2 was validated with respect to the CRAY version of CCM2 and a five-month simulation was performed successfully on the 512-processor Intel Delta computer. This is believed to be the first validated implementation of a global atmospheric circulation model on a scalable parallel computer.

Development of PCCM2 serves two objectives. First, it makes the computational capabilities of scalable parallel computers available to scientists using CCM2 for global change studies, allowing increases in spatial resolution, incorporation of improved process models, and longer simulations. For example, the 16 gigabytes of memory on a 128-processor IBM SP1 allows simulations at spectral resolutions greater than T200 (0.6 degrees by 0.6 degrees transform grid). Second, it provides a testbed that can be used to compare the performance of current scalable computers and conventional vector computers and to investigate issues such as load balancing, parallel I/O, and coupling that are important for future parallel models.

PCCM2 uses a message-passing, domain-decomposition approach, in which each processor is allocated responsibility for computation on one part of the computational grid, and messages are generated to communicate data between processors. Much of the research effort associated with development of a parallel code of this sort is concerned with identifying efficient decomposition and communication strategies. In PCCM2, this task is complicated by the need to support both semi-Lagrangian transport (for moisture) and spectral transport (for other fields). Load balancing and parallel I/O techniques are also required. In this paper, we review the various parallel algorithms used in PCCM2 and the work done to arrive at a validated model.

2. THE NCAR CCM

Over the past decade, the NCAR Climate and Global Dynamics Division has provided a comprehensive, three-dimensional global atmospheric model to university and NCAR scientists for use in the analysis and understanding of global climate. Because of its widespread use, the model was designated a Community Climate Model (CCM).

The most recent version of the CCM, CCM2, was released to the research community in October 1992 (Hack et al. 1992; Bath, Rosinski, and Olson 1992). This incorporates improved physical representations of a wide range of key climate processes, including clouds, radiation, moist convection, the planetary boundary layer, and transport. Changes to the pa-

*To appear in: *Proc. AMS Annual Meeting*, AMS, 1994.

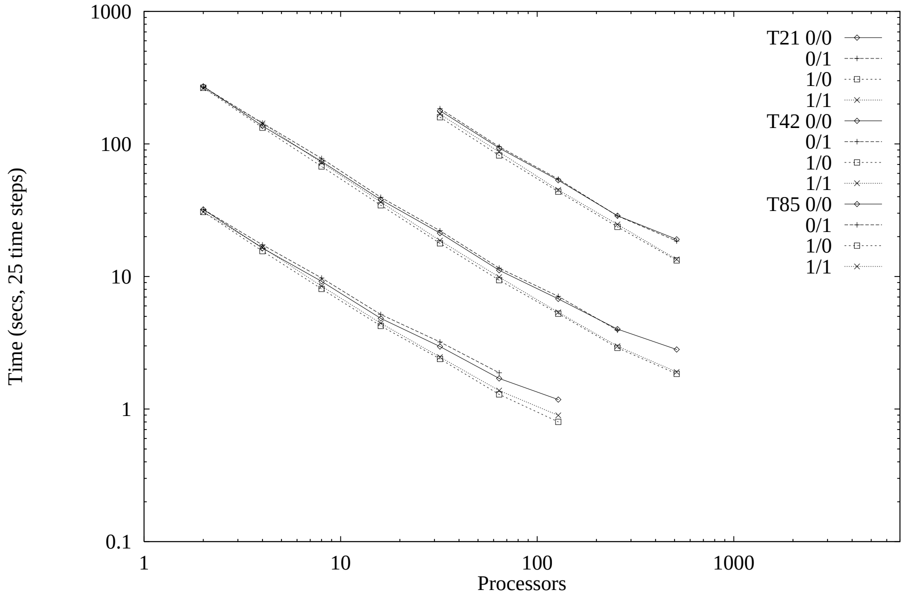

Figure 1: Shallow-water algorithm comparison on Intel Delta (see text for details)

rameterized physics include the incorporation of a diurnal cycle, along with the required multi-layer heat capacity soil model; major improvements to the radiation scheme, including a δ-Eddington solar scheme (18 spectral bands); a new cloud albedo parameterization; a new cloud emissivity formulation using liquid water path length; a new cloud fraction parameterization; and a Voigt correction to infrared radiative cooling in the stratosphere. Gravity-wave drag and explicit planetary boundary layer parameterizations are now included, and the moist adiabatic adjustment procedure has been replaced with a stability-dependent mass flux representation of moist convection.

The spherical harmonic (spectral) transform method is still used for the horizontal discretization of vorticity, divergence, temperature, and surface pressure, but a semi-Lagrangian transport scheme has been substituted for the advection of water vapor as well as for an arbitrary number of other scalar fields (e.g., cloud water variables, chemical constituents). The vertical coordinate now makes use of a hybrid (or generalized σ) formulation. The model has been developed for a standard horizontal spectral resolution of T42 (2.8 degrees by 2.8 degrees) with 18 vertical levels and a top at approximately 2.9 mb. The model code has also been entirely rewritten with three major objectives: greater ease of use and modification, conformation to a plug-compatible physics interface, and incorporation of single-job multitasking capabilities.

3. PARALLEL ALGORITHMS

CCM2 incorporates two major dynamics algorithms: the spectral transform method (Eliasen, Machenhauer, and Rasmussen 1970; Orszag 1970) and semi-Lagrangian transport (Williamson and Rasch 1989). The spectral transform method is used for the approximation of all terms in the equations except for the advective term in the moisture transport equation, which is updated using a semi-Lagrangian transport method. The spectral transform proceeds in two stages. First, the fast Fourier transform (FFT) is used to integrate information along each east-west gridline in the longitudinal direction, transforming physical space data to Fourier space. Second, a Legendre transform is used to integrate the results of the FFT in the north-south, or latitudinal, direction, transforming Fourier space data to spectral space. In the inverse transform, the order of these operations is reversed. In addition, numerous calculations are performed to simulate other physical processes such as clouds and radiation, moist convection, the planetary boundary layer, and surface processes. These other processes share the common feature that they are coupled horizontally only through the dynamics. These processes, collectively termed "physics," have the important property of being independent for each vertical column of grid cells in the model.

The parallel algorithms employed in PCCM2 are based on domain decomposition techniques. The three-dimensional data structures containing model variables are decomposed in two dimensions to obtain a number of contiguous blocks. Two such blocks,

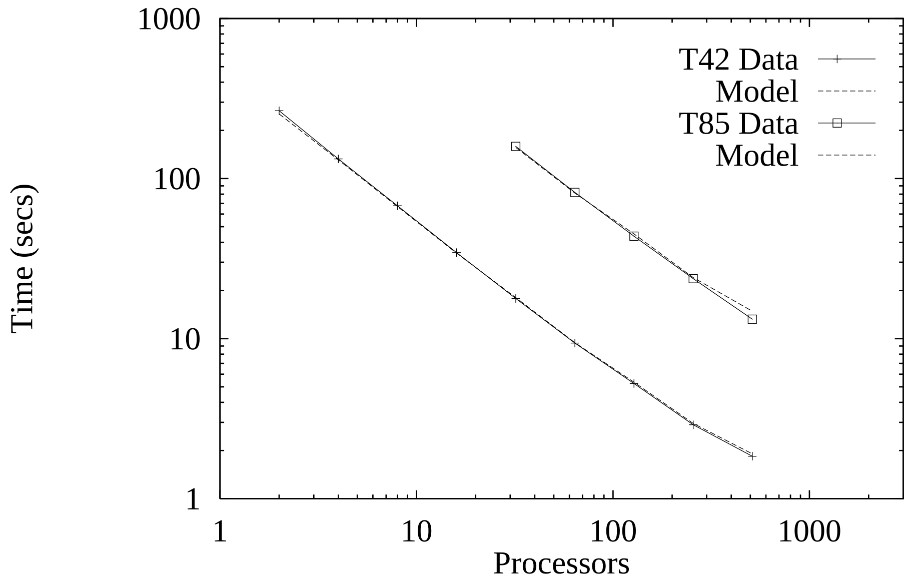

Figure 2: Modeled and observed execution times of shallow-water testbed on Intel Delta

one from the northern hemisphere and one from the southern hemisphere, are allocated to each processor, which performs the computation required for those blocks and communicates with other processors when data located in other blocks is required for a computation. Northern and southern latitudes are paired on processors to take advantage of symmetry in the spectral transform. Further details on the basic strategies and data structures employed have been provided in a previous paper (Drake et al. 1993).

A variety of different domain decompositions, and hence parallel algorithms, are possible (Foster and Worley 1993). Currently, PCCM2 uses a latitude/longitude decomposition of physical space, a latitude/vertical decomposition during FFTs, and a latitude/wavenumber decomposition of Fourier space. Spectral space, which is reached by a Gaussian approximation of a Legendre transform, is dimensioned M by N, where M is the highest Fourier wavenumber and N is the highest degree associated Legendre polynomial. Spectral space is decomposed in M only; the N dimension is replicated over processors. The physical space decomposition allows "physics" computations to proceed without communication within each vertical column, while the latitude/vertical decomposition allows the FFT to proceed without communication within each longitude. Communication is nevertheless required in two places. First, matrix transpose operations are required to move between the latitude/longitude decomposition employed in physical space, the latitude/wavenumber decomposition employed in Fourier space, and the latitude/vertical decomposition employed during Fourier transforms between the two spaces. Second, communication is required within the Gaussian approximation of the Legendre transform into spectral space. The latter operation is achieved using a parallel vector sum algorithm (Drake et al. 1993); other algorithms have been found to be slightly more efficient in some cases, but are harder to integrate into PCCM2 (Worley and Drake 1992).

The semi-Lagrangian transport method used in CCM2 updates the value of the moisture field at a grid point (the arrival point, A) by first establishing a trajectory through which the particle arriving at A has moved during the current timestep. From this trajectory the departure point, D, is calculated, and the moisture field is interpolated at D using shape preserving interpolation. Since the semi-Lagrangian transport occurs entirely in physical space, these calculations are decomposed by using the latitude/longitude decomposition used to parallelize the physics and spectral transform. Communication is then required whenever D and A are on different processors. In the current implementation of PCCM2, this communication is achieved by extending the arrays on each processor so that "overlapping" regions are assigned to neighboring processors. Overlapped portions of the array are updated prior to each timestep via

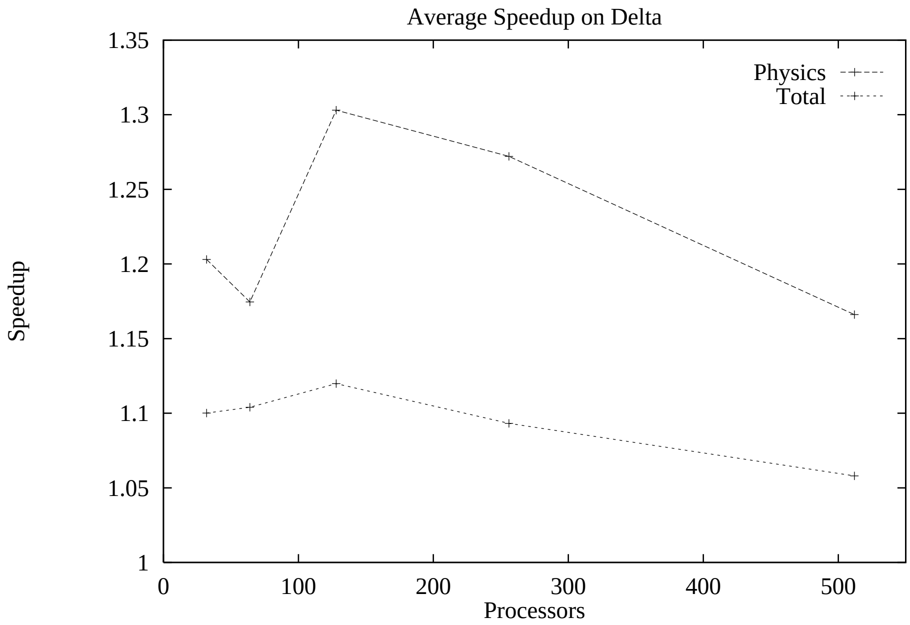

Figure 3: Speedups from load balancing

interprocessor communication, so that calculations on the different processors can proceed independently. This strategy has the advantage of simplicity. However, it is computationally inefficient, as the distance travelled (and hence the required overlap region) can be as high as 16 grid points near the poles at T42 resolution. Improved parallel algorithms that reduce communication requirements are a topic of current research.

4. ALGORITHM COMPARISONS

The domain decompositions and parallel algorithms employed in PCCM2 are not the only ones possible, nor necessarily the most efficient. Hence, empirical and analytic studies have been conducted to provide a detailed understanding of performance issues in various parallel algorithms (Foster and Worley 1993). The empirical studies utilize a parallel shallow-water equation solver designed specifically for these experiments. This code is derived from the sequential Fortran code STSWM developed for numerical studies of the shallow water equations (Hack and Jackob 1992). It incorporates a wide variety of algorithm variants in a modular fashion. Care has been taken to ensure that experiments are as fair as possible; that is, that one algorithm is not unduly favored through choice of data structures, greater optimization, etc. In addition, the code structure has been designed to mimic that utilized in general circulation models, so as to maximize the applicability of results to these models. In particular, it has been extended with vertical levels.

Preliminary results suggest that the optimal parallel algorithm for computers such as the Intel Paragon may employ a latitude/longitude decomposition of physical space, a latitude/vertical decomposition for FFTs, and a N/vertical decomposition of spectral space and may, furthermore, employ a recursive summation algorithm for the approximation of the Legendre transform (LT). Like PCCM2, this algorithm requires a matrix transpose; however, the use of an N/vertical decomposition rather than an N/M decomposition in spectral space halves the number of transpose operations required.

Figure 1 shows time required for 25 time steps as a function of processor count on the 512-processor Intel Delta, for four different parallel algorithms and three different spectral truncations. (Notice the use of log scales.) The three sets of curves are, from top to bottom, for T85, T42, and T21 resolution. The parallel algorithms studied are parallel FFT/recursive summation LT (0/0); transpose FFT/transpose LT (1/1); parallel FFT/transpose LT (0/1); and transpose FFT/recursive summation LT (1/0). Algorithms 1/0 and 1/1 are clearly superior to algorithms 0/0 and 0/1, and algorithm 1/0 is

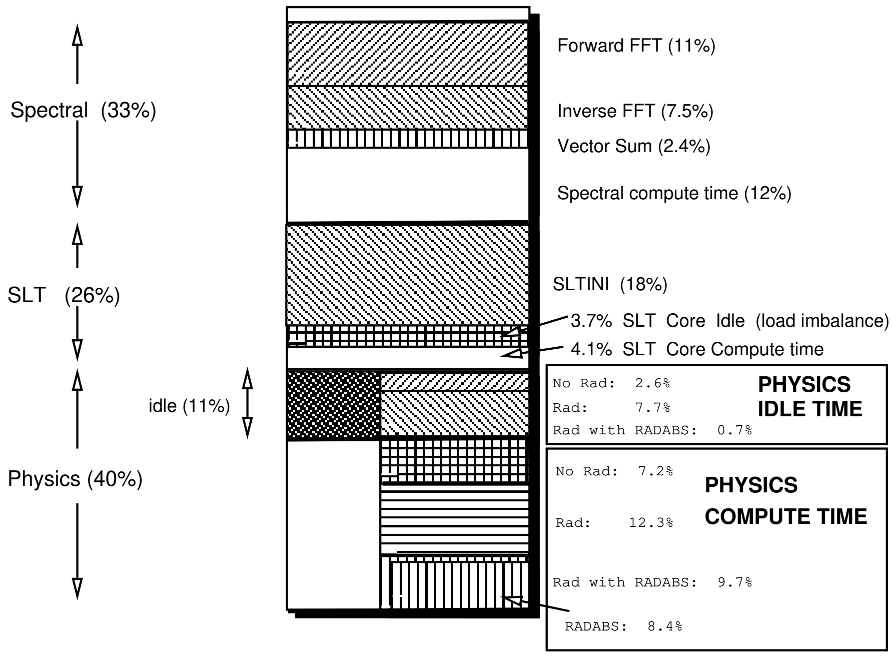

Figure 4: PCCM2 performance on 512 nodes of the Intel Delta, showing detailed breakdown of time spent within the model. Interprocess communication in forward and inverse Fourier transforms, as well as in initialization of extended arrays in the semi-Lagrangian transport (SLTINI), consumes a significant portion of total time

slightly faster than algorithm 1/1.

Results also suggest that the optimal parallel algorithm depends on both problem size and machine characteristics. Hence, performance studies are continuing to investigate different algorithm benefits and different machines, including the Intel Paragon and IBM SP1.

Analytic studies have been conducted that use performance modeling techniques to relate observed performance to algorithm and machine characteristics (Foster, Gropp, and Stevens 1992; Foster and Worley 1993). Good fits between model and observed execution times have been observed, as illustrated in Figure 2 which shows results for the transpose FFT/recursive summation LT algorithm at T42 and T85 resolution. Notice that the model works well except at T85 resolution on 512 processors, where predicted times are too high. The development of these analytic models allows the evaluation of performance tradeoffs on different parallel computer architectures.

5. OTHER ISSUES

Although the development of efficient parallel spectral transform and semi-Lagrangian transport algorithms has been a major focus of this project, the development of PCCM2 has also required the development of a number of other techniques. These include the development of efficient global summation algorithms, non-power-of-two FFTs, parallel I/O techniques, and load-balancing algorithms. Work on the latter two topics is summarized here.

As parallel computers are used to execute climate models faster, the rate at which data is generated also increases. For example, PCCM2 currently executes at a rate of about one simulation day per minute on the Intel Delta. If diagnostic data (a “history tape” record) is to be recorded once every twelve hours for diagnostic purposes, then clearly the output of this data to disk cannot take more than a few seconds if it is not to be a serious bottleneck. A history tape record is about 6 megabytes (MB) at T42 resolution, so burst bandwidth of at least 2 MB/sec is required. As model performance improves, bandwidth requirements will increase.

Fortunately, parallel computers also provide a natural mechanism for improving I/O performance: different processors can perform output operations concurrently, writing different data to different disks. However, this requires the development of specialized parallel libraries both to write data in this fashion and to read the data once it has been written. Libraries of this sort have been developed for the Intel Delta. These write a T42 restart dataset of 30 MB in approximately 4 seconds, which corresponds to a transfer rate above 7 MB/sec. Using asynchronous write on the Delta reduces the effective cost to on the order of several milliseconds, the time needed to initiate the write request. Afterwards, the program continues computing while the output operation completes. The algorithms are currently being refined and adapted for the Intel Paragon and IBM SP1.

Parallel I/O is complicated by the fact that the program that reads an output data set may not run on the same number of processors, or even on the same computer. The I/O libraries developed for PCCM2 allow for this. For example, in a recent experiment, a four-month simulation was conducted on 256 processors of the Intel Delta, and then restarted and continued for two additional months on 4 processors of an IBM SP1.

Load imbalances can arise in parallel climate models as a result of spatial and temporal variations in the computation requirements of "physics" routines (Michalakes 1990). For example, CCM2 performs radiation computations only in grid points that are in sunlight. This situation can be corrected by employing load-balancing algorithms that, either statically or dynamically, map computation to processors in different ways.

A flexible load-balancing library has been developed as part of the PCCM2 effort, suitable for use in PCCM2 and other similar climate models. This library has been incorporated into PCCM2 and used to experiment with alternative load-balancing strategies. One simple strategy swaps every second grid point with its partner 180 degrees apart in longitude. This does a good job of balancing the diurnal cycle: because CCM2 pairs N/S latitudes, it ensures that each processor always has approximately equal numbers of day and night points. However, it generates a lot of communication. Another strategy uses a dynamic data-distribution strategy that performs less communication but requires periodic reorganizations of model state data. Currently, the former scheme gives better performance in most cases, and succeeds in eliminating about 75 per cent of the inefficiency attributable to load imbalances when PCCM2 runs on 512 Intel Delta processors. The second scheme is expected to become more competitive with further tuning.

6. PARALLEL PERFORMANCE

Performance studies in early 1993 revealed that PCCM2 achieved about 750 Mflops on 512 processors of the Intel Delta at T42 resolution. Figure 4 shows where time was spent in this 750 Mflops run. Parallel efficiency is about 25 per cent; the rest of the time is spent in communication, particularly in the FFT and SLT routines, and in idle time becuase of load imbalances in physics. Since these results were obtained, performance has been improved by the incorporation of load-balancing algorithms and a transpose FFT. On the other hand, the size of the semi-Lagrangian transport overlap regions has been increased, which has reduced performance.

For comparison, CCM2 executes at just over 1 Gflops for CCM2 on a CRAY Y-MP/8 and significantly faster on a CRAY C90. Clearly, PCCM2 performance is creditable but not yet competitive with vector supercomputers. Preliminary experiments suggest that performance should improve significantly on the Intel Paragon and IBM SP1 to which PCCM2 is being ported at ORNL and ANL, respectively. In addition, various aspects of the parallel code are being refined with a view to improving performance on these computers. The improved SLT, load balancing, and parallel I/O algorithms referred to previously are all part of this work. Another is the blocking of FFTs so as to reduce the number of communication operations. At T42 resolution and on 64 Intel Delta processors, this reduces time spent in FFTs by 34 per cent and total execution time by 5 per cent.

7. VALIDATION

The first simulations with PCCM2 were performed in late 1992. These gave what appeared to be realistic results. Detailed validation studies were then undertaken in order to verify correctness. Simulations performed on an IEEE-compliant microprocessor such as the Intel i860 or the IBM RS6000 cannot be expected to be identical to those performed on a CRAY: differences in machine arithmetic result in low-order divergences, which grow slowly over time. Nevertheless, it is important to distinguish these acceptable differences from subtle errors caused by algorithmic errors or insufficient machine precision, since the latter may produce incorrect climate statistics. The strategy employed was to study the rate at which PCCM2 and the original model diverged, as measured by differences in RMS values of output fields such as temperature, vorticity, divergence, and pressure and to compare this divergence rate with that observed when the original model's input data was perturbed in the least significant bit. Figure 5 shows two plots, the first of which shows error growth in the sequential model when a minimal perturbation is introduced. The second plot shows error growth comparing a run of the sequential model with a run of the parallel model. To be considered a valid port, the separation of the parallel model from the original should grow no faster than the minimal perturbation error grows in the original code. This was the case

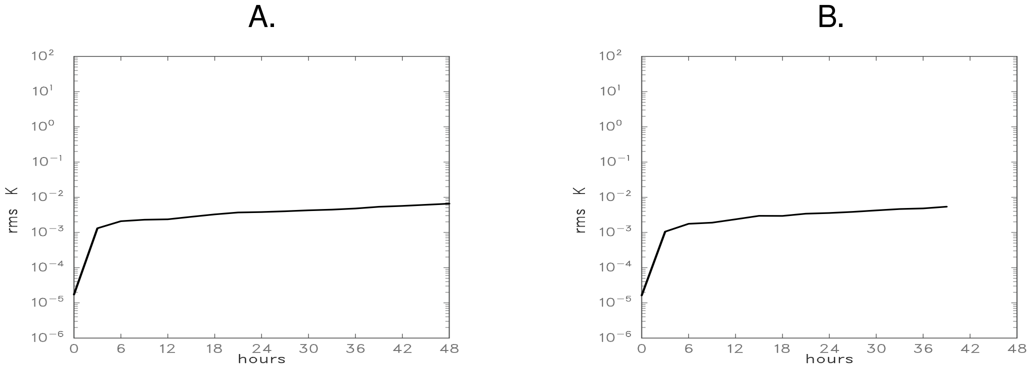

Figure 5: Error growth in temperature field: *A* shows growth comparing runs of sequential code with a 1-bit perturbance; *B* shows growth comparing the parallel code with the original code.

for single and double precision on the IBMs and in single precision on the Intel Delta (double has not been tested).

Validation of PCCM2 was simplified by the fact that the code had been designed to give identical results independent of the number of processors on which it was running. This uniformity is not straightforward to achieve, because of the lack of associativity in floating-point addition, but was obtained in PCCM2 by modifying the sequential implementation of various summation operations to use the same tree-based summation algorithm as the parallel code. This work reduced the validation task to studying error divergence with a fixed number of processors and to verifying that the results computed by the parallel model were identical for all processor counts. A related benefit of this "reproducibility" is that PCCM2 simulations can be restarted with varying numbers of processors without affecting the numbers computed.

The validation studies indicated that it was acceptable to perform the majority of the model calculations in single-precision (32-bit) arithmetic. Only the Gaussian weights, sines of latitude, and associated Legendre polynomials need to be calculated in double precision (and then truncated to single precision). The validation studies also uncovered two subtle errors in the parallel implementation. It seems unlikely that these errors would have been detected without a careful study of divergence rates.

8. SUMMARY

PCCM2 is now operational on scalable parallel computers and can be used for scientific studies. It has successfully completed five-month simulations and incorporates the history tape output and restart capabilities required for long simulations. Its performance on the 512-processor Intel Delta is comparable to that of CCM2 on a CRAY Y-MP; preliminary results suggest that performance on the Intel Paragon and IBM SP1 should be considerably better.

In future work, PCCM2 will be optimized for more efficient execution on scalable parallel computers. The goal of this work is to produce a climate model that outperforms the fastest vector supercomputers.

ACKNOWLEDGMENTS

This research was supported by the DOE CHAMMP initiative. Access to the Intel Delta was provided by the Concurrent Supercomputing Consortium.

REFERENCES

BATH, L. J., J. ROSINSKI, AND J. OLSON, 1992: *Users' Guide to the NCAR CCM2*, NCAR Technical Note NCAR/TN-379+IA, National Center for Atmospheric Research, Boulder, Colo.

DEPARTMENT OF ENERGY, 1990: *Building an advanced climate model: Progress plan for the CHAMMP climate modeling program*, DOE Tech. Report DOE/ER-0479T, U.S. Department of Energy, Washington, D.C., December.

DRAKE, J. B., R. E. FLANERY, I. T. FOSTER, J. J. HACK, J. G. MICHALAKES,

R. L. STEVENS, D. W. WALKER, D. L. WILLIAMSON, AND P. H. WORLEY, 1993: *The message passing version of the Parallel Community Climate Model*, in Parallel Supercomputing in Atmospheric Science, G.-R. Hoffmann and T. Kauranne, eds., World Scientific, Singapore, 500–513.

ELIASEN, E., B. MACHENHAUER, AND E. RASMUSSEN, 1970: *On a numerical method for integration of the hydrodynamical equations with a spectral representation of the horizontal fields*, Report No. 2, Institut for Teoretisk Meteorologi, Kobenhavns Universitet, Denmark.

FOSTER, I., W. GROPP, AND R. STEVENS, 1992: The parallel scalability of the spectral transform method, *Mon. Wea. Rev.*, 125, 835–850.

FOSTER, I. T., AND P. H. WORLEY, 1993: *Parallelizing the spectral transform method: A comparison of alternative parallel algorithms*, in Parallel Processing for Scientific Computing, R. F. Sincovec, D. E. Keyes, M. R. Leuze, L. R. Petzold, and D. A. Reed, eds., Society for Industrial and Applied Mathematics, Philadelphia, Penna., 100–107.

HACK, J. J., B. A. BOVILLE, B. P. BRIEGLEB, J. T. KIEHL, P. J. RASCH, AND D. L. WILLIAMSON, 1992: *Description of the NCAR Community Climate Model (CCM2)*, NCAR Technical Note TN-382+STR, National Center for Atmospheric Research, Boulder, Colo.

HACK, J. J., AND R. JACKOB, 1992: *Description of a global shallow water model based on the spectral transform method*, NCAR Technical Note TN-343+STR, National Center for Atmospheric Research, Boulder, Colo.

MICHALAKES, J., 1991: *Analysis of workload and load balancing issues in NCAR Community Climate Model*, Technical Report ANL/MCS-TM-144, Argonne National Laboratory, Argonne, Ill.

ORSZAG, S. A., 1970: Transform method for calculation of vector-coupled sums: Application to the spectral form of the vorticity equation, *J. Atmos. Sci.*, 27, 890–895.

WILLIAMSON, D. L., AND P. J. RASCH, 1989: Two-dimensional semi-Lagrangian transport with shape-preserving interpolation, *Mon. Wea. Rev.*, 117, 102–129.

WORLEY, P. H., AND J. B. DRAKE, 1992: Parallelizing the spectral transform method, *Concurrency: Practice and Experience*, 4, 269–291.