Performance engineering in the community atmosphere model

Institute of Physics Publishing Journal of Physics: Conference Series **46** (2006) 356–362
doi:10.1088/1742-6596/46/1/050 SciDAC 2006

Performance engineering in the community atmosphere model

P Worley[1], A Mirin[2], J Drake[1] and W Sawyer[3]

[1] Oak Ridge National Laboratory, P.O. Box 2008, Oak Ridge, TN 37831-6016
[2] Lawrence Livermore National Laboratory, Livermore, CA 94551
[3] Swiss Federal Institue of Technology, 8092 Zurich, Switzerland

E-mail: worleyph@ornl.gov

Abstract.

The Community Atmosphere Model (CAM) is the atmospheric component of the Community Climate System Model (CCSM) and is the primary consumer of computer resources in typical CCSM simulations. Performance engineering has been an important aspect of CAM development throughout its existence. This paper briefly summarizes these efforts and their impacts over the past five years.

1. Introduction

The Community Climate System Model (CCSM) is a fully-coupled, global climate model that provides state-of-the-art computer simulations of the Earth's past, present, and future climate states [1]. CCSM development is the primary activity of the Department of Energy (DOE) Scientific Discovery through Advanced Computing (SciDAC) project *Collaborative Design and Development of the Community Climate System Model for Terascale Computers*. Performance and performance portability have been major foci of this project since its inception in July 2001.

CCSM is constantly evolving to incorporate new science. The target computer platforms change periodically as well. The project performance engineering goals are addressed by an iterative process that includes porting CCSM to new platforms, collecting and analyzing performance data, optimizing performance, and modifying code to improve performance portability. Performance portability refers to the capability of a code to be optimized on a new platform or for a new problem instance quickly, and is an important enabler of this iterative process. These performance engineering activites are performed in close collaboration with the SciDAC Integrated Software Infrastructure Center on performance engineering *Performance Evaluation Research Center*.

CCSM is made up of a coupler and four component models: atmosphere, ocean, land, and sea ice. The Community Atmosphere Model (CAM) is the atmospheric component of the CCSM and is the primary consumer of computer resources in typical CCSM simulations. This paper briefly summarizes the impact of the above mentioned SciDAC projects on both performance and performance portability of CAM over the past five years. For information on performance engineering in other CCSM component models, see the special issue of the International Journal on High Performance Computer Applications (2005, volume 19, number 3) on climate modeling.

2. Community Atmosphere Model

CAM is a global atmosphere model developed at the National Science Foundation's National Center for Atmospheric Research (NCAR) with contributions from researchers funded by DOE and by the National Aeronautics and Space Administration (NASA) [2]. CAM is a mixed-mode parallel application code, using both the Message Passing Interface (MPI) [6] and OpenMP protocols [5]. CAM is characterized by two computational phases: the dynamics, which advances the evolution equations for the atmospheric flow, and the physics, which approximates subgrid phenomena such as precipitation processes, clouds, long- and short-wave radiation, and turbulent mixing [3]. The approximations in the physics are referred to as the physical parameterizations. Control moves between the dynamics and the physics at least once during each model simulation timestep.

CAM includes multiple options for the dynamics, referred to as *dynamical cores* or *dycores*, one of which is selected at compile-time. Three dycores are currently supported: a spectral Eulerian (EUL) [7], a spectral semi-Lagrangian (SLD) [12], and a finite volume semi-Lagrangian (FV) [8]. The spectral and finite volume dycores use different computational grids. An explicit interface exists between the dynamics and the physics, and the physics data structures and parallelization strategies are independent from those in the dynamics. A dynamics-physics coupler moves data between data structures representing the dynamics state and the physics state.

The development of CAM is a large community-wide effort, with software engineering led by the CCSM Software Engineering Group at NCAR. The authors were instrumental in much of the performance engineering of CAM [9, 14], including all three dycores and the physics. However, some of the performance portability features in the FV dycore were developed at NASA [10] and David Parks of NEC Solutions America, in partnership with the Japanese Earth Simulator Center, was responsible for the initial vectorization of many of the routines in CAM.

3. Performance Engineering

The general performance engineering goals are to (1) maximize single processor performance, e.g., optimize memory access patterns and maximize vectorization or other fine-grain parallelism, and (2) minimize parallel overhead, e.g., minimize communication costs, load imbalance, and redundant computation. These goals need to be achieved for a variety of target computer systems, a range of problem specifications, and a range of processor counts, all while preserving maintainability and extensibility. There is no optimal solution for all desired configurations of platform, problem, and processor count, and we rely on performance portability techniques. We have been successful in delaying decisions that affect performance until compile-time or run-time by hiding performance options in utility layers or in initialization routines. The code seen by developers has not been significantly impacted. The current set of CAM tuning options are discussed in [13].

Most of our optimization efforts can be categorized as either (1) eliminating unnecessary work, (2) cache blocking and/or vectorization, (3) exposing additional parallelism, (4) load balancing, (5) interprocessor communication optimizations, or (6) evaluating different compiler optimization options and exploiting optimized libraries. The largest impact on the code, in terms of number of lines modified, has come from exposing additional parallelism, e.g., by introducing a two-dimensional domain decomposition where before there was only a one-dimensional decomposition, and from enabling vectorization, e.g., by reordering loops and changing data structures to increase loop lengths.

4. Experimental Platforms

Performance data are presented for a number of different high performance computing systems, as described below.

- Cray X1 at Oak Ridge National Laboratory (ORNL): 512 Multi-Streaming Processors (MSP), each capable of 12.8 GFlop/s for 64-bit operations. MSPs are fully connected within 16-MSP subsets, and are connected via a 2-D torus between subsets.
- Cray X1E at ORNL: 1024 MSPs, each capable of 18 GFlop/s for 64-bit operations. MSPs are fully connected within 32-MSP subsets, and are connected via a 2-D torus between subsets. This system is an upgrade of the original Cray X1 at ORNL.
- Cray XT3 at ORNL: 5294 single processor nodes (2.4 GHz AMD Opteron) and a 3-D torus interconnect. Each processor is capable of 4.8 GFlop/s for 64-bit operations.
- Earth Simulator: 640 8-way symmetric multiprocessor (SMP) nodes and a 640x640 single-stage crossbar interconnect. Each processor has 8 64-bit floating point vector units running at 500 MHz, and is capable of 8 GFlop/s for 64-bit operations.
- IBM p575 cluster at the National Energy Research Scientific Computing Center (NERSC): 122 8-way p575 SMP nodes (1.9 GHz POWER5 processors) and an HPS interconnect with 1 two-link adapter per node. Each processor is capable of 7.6 GFlop/s for 64-bit operations.
- IBM p690 cluster at ORNL: 27 32-way p690 SMP nodes (1.3 GHz POWER4 processors) and an HPS interconnect. Each node has two HPS adapters, each with two ports. Each processor is capable of 5.2 GFlop/s for 64-bit operations.
- IBM SP at NERSC: 184 Nighthawk II 16-way SMP nodes (375 MHz POWER3-II processors) and an SP Switch2. Each node has two interconnect interfaces. Each processor is capable of 1.5 GFlop/s for 64-bit operations.
- Itanium2 cluster at Lawrence Livermore National Laboratory: 1024 4-way Tiger4 nodes (1.4 GHz Intel Itanium 2) and a Quadrics QsNetII Elan4 interconnect. Each processor is capable of 5.6 GFlop/s for 64-bit operations.
- SGI Origin 3800 at NASA Ames Research Center: 1024 600 MHz MIPS R14000 processors and a NUMAlink switch that supports nonuniform memory access global shared memory. Each processor is capable of 1.2 GFlop/s for 64-bit operations.

5. Performance History

Documenting performance improvements in an evolving code can be difficult. Changes that add, for example, new processes to the physics can increase the amount of work required in a simulation. Such complexity-changing modifications invalidate performance optimization comparisons between versions of the code before and after the modification. Periodically the problem specification or dynamical core of most interest also changes. It is often not possible to run the new problem specification using older versions of the code, requiring the redefinition of baseline performance using the current production version of the code. A version control system is essential for this type of study as it documents the code evolution. Subversion [4] is used with CAM currently, having replaced CVS in December 2005. One positive side effect of introducing performance optimizations as compile-time and run-time options is that new versions of the code can be run in the "old way", so that the performance impact of the introduction of a performance tuning option can still be quantified.

The left graph in Figure 1 describes the performance improvement between June 2001 and November of 2002 on the IBM p690 cluster. During this time period the p690 cluster had an IBM SP Switch2 interconnect between nodes. The benchmark problem T42 L26 uses a 64x128x26 (latitude by longitude by vertical) computational grid and the EUL dycore, which were the production settings at the time of the experiments. The "original settings" curve is the performance when setting the optimization options to emulate the way that CAM version 1.0 was run in June 2001. Note that the performance improvements include increasing the number of processors that could be used effectively from 64 to more than 256.

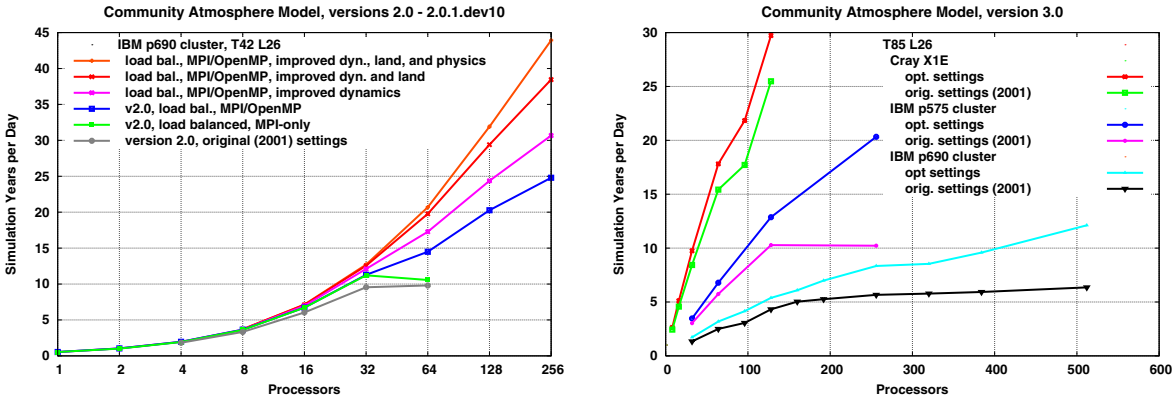

Figure 1. CAM EUL Performance History.

The right graph in Figure 1 describes the performance improvement attributable to the performance optimization options introduced between June 2001 and May 2004 on the IBM p690 cluster, now using the IBM HPS interconnect, on the IBM p575 cluster, and on the Cray X1E. Results are for the new production problem size of T85 L26, which uses the EUL dycore and a computational grid of size 128x256x26.

So far we have described the impact of performance optimizations in the physics and in the EUL dycore. Over the same period of time the NASA FV dycore was integrated with CAM, a two-dimensional (2D) domain decomposition option was implemented, and a number of FV communication optimization options were added. FV was not available in CAM prior to this work and many of the optimizations were introduced simultaneously. In consequence, no baseline exists against which to evaluate these optimizations. Figure 2 contains plots examining CAM performance sensitivities to two of the FV-specific tuning options.

The left graph in Figure 2 compares CAM performance when using MPI-2 one-sided and MPI-1 two-sided communication requests with varying levels of thread parallelism. The experimental platform is the SGI Origin3800. The benchmark problem (1x1.25 L26) uses a 181x288x26 computational grid. Note that MPI-2 one-sided communications and thread level parallelism are not performance enhancers on all target platforms, and it is important for performance portability that the communication protocol be an option.

The right graph in Figure 2 illustrates the performance impact of the 2D domain decomposition on performance on three platforms: the Cray X1E, the Cray XT3, and the IBM p690 cluster. The benchmark problem (0.5x0.625 L26) uses a 361x576x26 computational grid. Performance is graphed for the original one-dimensional (1D) decomposition, a 2D decomposition where four processes are applied to the new dimension, and a 2D decomposition in which seven processes are applied to the new dimension. The 1D decomposition is limited to 120 MPI processes for this benchmark. The 2D domain decomposition increases MPI scalability significantly, but with diminishing performance returns for high processor counts. For the IBM p690 cluster, the 1D domain decomposition with OpenMP parallelism is more efficient at increasing scalability than is the 2D decomposition with OpenMP parallelism up to the indicated number of processors.

The most recent performance engineering efforts involved (re)vectorizing CAM while not degrading performance on the nonvector systems. The target vector systems are the Cray X1E and the Earth Simulator. These two systems have somewhat different performance sensitivities, and maintaining performance portability between them has required careful testing. Figure 3 illustrates the performance history for three benchmark problems and processor counts on the X1E from May 2005 to May 2006. The versions named on the X-axis were developed on the

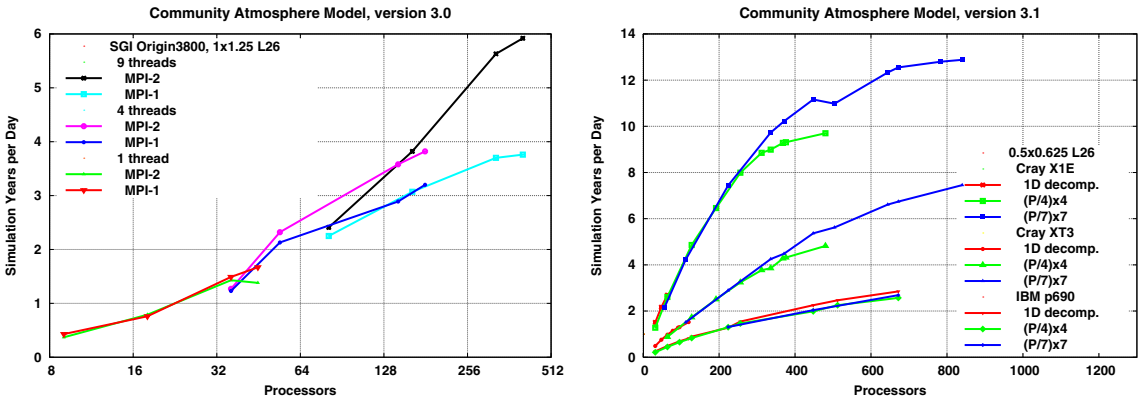

Figure 2. CAM FV Performance Optimizations.

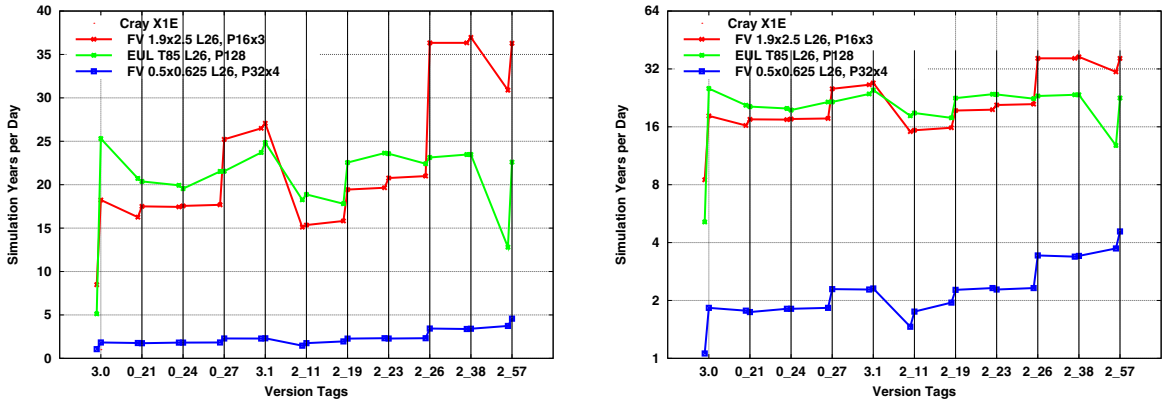

Figure 3. CAM Vector Performance History: March 2005 to March 2006.

X1E, and often include changes that eliminate performance degradations that crept in since the previous X1E-oriented modification. For each named version, we also measured performance for the immediately preceding version. The name of each version is of the form "3.X_Y". The "3." is dropped from the name in the graph where it improves readability. The two graphs contain the same data, but the one on the right uses a logarithmic Y-axis. From this it should be clear that maintaining vectorization is a significant performance advantage, and requires ongoing monitoring as new code is introduced. The new benchmark problem, 1.9x2.5 L26, has a 96x144x26 computational grid and is the initial production resolution for the FV dycore within CCSM.

6. Future Challenges

As shown in Section 5, performance engineering efforts over the past five years have resulted in significant performance improvements in CAM. The graphs in Figure 4 describe current CAM performance, where recent performance optimizations have been backported into versions 3.0 and 3.1. While performance on the current production platforms is very good, scalability is still limited. CAM is evolving quickly. It will soon include comprehensive treatments of the processes governing well-mixed greenhouse gases, natural and anthropogenic aerosols, the aerosol indirect effect and trapospheric ozone for climate change studies. These improvements come at the cost of significant increases in computational complexity, minimally 3-5 times as costly per vertical column. A vertical column refers to to all grid points with a given horizontal

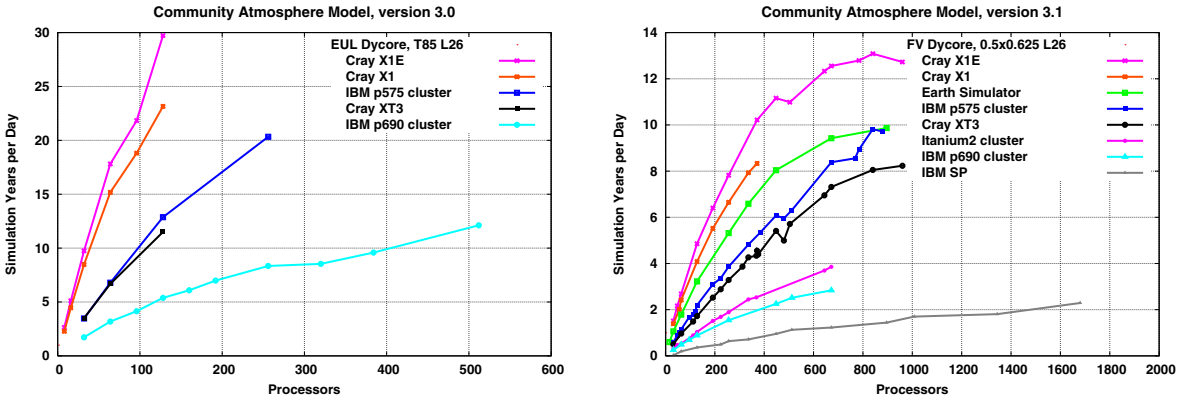

Figure 4. CAM Performance, May 2006

location, differing only in the vertical coordinate. In order to maintain the required simulation throughput rates, we are investigating whether additional domain decompositions with respect to chemical constituent and the vertical coordinate can be used to improve scalability. We are also examining the performance and scalability of new dycores utilizing computational grids, such as the cubed sphere [11], that avoid the polar singularity of the traditional latitude-longitude approach and lend themselves more naturally to 2-D and 3-D domain decompositions. We will continue to address load imbalance in the physics and communication overhead in the dynamics as the new processes are incorporated into CAM. We will also continue to emphasize algorithmic flexibility and peformance portability in our work, believing these to be vital to achieving CAM performance goals.

Acknowledgments

We thank D. Parks of NEC Solutions America for providing CAM performance data on the Earth Simulator, and M. Wehner of Lawrence Berkeley National Laboratory for providing CAM performance data on the IBM SP. We also gratefully acknowledge our collaborators, too numerous to mention here, in the performance engineering of CAM.

This research used resources (Cray X1E, Cray XT3, IBM p690 cluster) of the National Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. It used resources (IBM p575 cluster) of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

The work of Drake and Worley was supported by the Climate Change Research Division of the Office of Biological and Environmental Research and by the Office of Mathematical, Information, and Computational Sciences, both in the Office of Science, U.S. Department of Energy, under Contract No. DE-AC05-00OR22725 with UT-Batelle, LLC. The work of Mirin was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48, and this paper is LLNL report UCRL-CONF-221711. The work of Sawyer was performed under the auspices of the NASA Goddard Space Flight Center, NASA Task 00-9103-01a. Accordingly, the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

References

[1] M. B. Blackmon, B. Boville, F. Bryan, R. Dickinson, P. Gent, J. Kiehl, R. Moritz, D. Randall, J. Shukla, S. Solomon, G. Bonan, S. Doney, I. Fung, J. Hack, E. Hunke, and J. Hurrel, *The Community Climate System Model*, BAMS, 82 (2001), pp. 2357–2376.

[2] W. D. Collins, P. J. Rasch, B. A. Boville, J. J. Hack, J. R. McCaa, D. L. Williamson, B. P. Briegleb, C. M. Bitz, S.-J. Lin, and M. Zhang, *The Formulation and Atmospheric Simulation of the Community Atmosphere Model: CAM3*, Journal of Climate, 19(11) (2006).

[3] W. D. Collins, P. J. Rasch, and et. al., *Description of the NCAR Community Atmosphere Model (CAM 3.0)*, NCAR Tech Note NCAR/TN-464+STR, National Center for Atmospheric Research, Boulder, CO 80307, 2004.

[4] B. Collins-Sussman, B. W. Fitzpatrick, and C. M. Pilato, *Version Control with Subversion*, O'Reilly Media, Inc., Sebastopol, CA, June 2004.

[5] L. Dagum and R. Menon, *OpenMP: an industry-standard API for shared-memory programming*, IEEE Computational Science & Engineering, 5 (1998), pp. 46–55.

[6] W. Gropp, M. Snir, B. Nitzberg, and E. Lusk, *MPI: The Complete Reference*, MIT Press, Boston, 1998. second edition.

[7] J. T. Kiehl, J. J. Hack, G. Bonan, B. A. Boville, D. L. Williamson, and P. J. Rasch, *The National Center for Atmospheric Research Community Climate Model: CCM3*, J. Climate, 11 (1998), pp. 1131–1149.

[8] S.-J. Lin, *A 'vertically Lagrangian' finite-volume dynamical core for global models*, Mon. Wea. Rev., 132 (2004), pp. 2293–2307.

[9] A. Mirin and W. B. Sawyer, *A scalable implemenation of a finite-volume dynamical core in the Community Atmosphere Model*, International Journal of High Performance Computing Applications, 19 (2005), pp. 203–212.

[10] W. Putman, S. J. Lin, and B. Shen, *Cross-platform performance of a portable communication module and the NASA finite volume general circulation model*, International Journal of High Performance Computing Applications, 19 (2005), pp. 213–224.

[11] M. Rancic, R. J. Purser, and F. Mesinger, *A global shallow water model using an expanded spherical cube: gnomonic versus conformal coordinates*, Quart. J. Roy. Meteor. Soc., 122 (1996), pp. 959–982.

[12] D. L. Williamson and J. G. Olson, *Climate simulations with a semi-lagrangian version of the NCAR Community Climate Model*, Mon. Wea. Rev., 122 (1994), pp. 1594–1610.

[13] P. H. Worley, *Benchmarking using the Community Atmosphere Model*, in Proceedings of the 2006 SPEC Benchmark Workshop, Jan. 23, 2006, Warrenton, VA, 2006, The Standard Performance Evaluation Corp.

[14] P. H. Worley and J. B. Drake, *Performance portability in the physical parameterizations of the Community Atmosphere Model*, International Journal of High Performance Computing Applications, 19 (2005), pp. 187–202.