

OPEN ACCESS

Extending scalability of the community atmosphere model

To cite this article: A Mirin and P Worley 2007 *J. Phys.: Conf. Ser.* **78** 012082

View the [article online](#) for updates and enhancements.

Related content

- [Performance engineering in the community atmosphere model](#)
P Worley, A Mirin, J Drake et al.
- [Petascale atmospheric models for the Community Climate System Model: new developments and evaluation of scalable dynamical cores](#)
M A Taylor, J Edwards and A St Cyr
- [Incomplete Balancing Domain Decomposition for Large scale 3-D non-stationary incompressible flow problems](#)
Q H Yao, H Kanayama, M Ognio et al.

Recent citations

- [Evaluation of a CCSM3 Simulation with a Finite Volume Dynamical Core for the Atmosphere at 1° Latitude × 1.25° Longitude Resolution](#)
G. Bala *et al*



240th ECS Meeting

Digital Meeting, Oct 10-14, 2021

We are going fully digital!

Attendees register for free!

REGISTER NOW



Extending Scalability of the Community Atmosphere Model

A Mirin¹ and P Worley²

¹ Lawrence Livermore National Laboratory, Livermore, CA 94551

² Oak Ridge National Laboratory, P.O. Box 2008, Oak Ridge, TN 37831-6016

E-mail: worleyph@ornl.gov

Abstract.

The Community Atmosphere Model (CAM) is the atmosphere component of the Community Climate System Model (CCSM), and is the largest consumer of computing resources in typical CCSM simulations. The parallel implementation of the Community Atmosphere Model (CAM) employs a number of different domain decompositions. Currently, each decomposition must utilize the same number of MPI processes, limiting the scalability of CAM to that of the least scalable decomposition. This limitation becomes unacceptably restrictive when including additional physical processes such as atmospheric chemistry or cloud-resolving physics. This paper reports on current efforts to improve CAM scalability by allowing the number of active MPI processes to vary between domain decompositions.

1. Introduction

The Community Climate System Model (CCSM) [6, 3] is one of the primary tools for climate science studies in the United States. The CCSM consists of atmosphere, ocean, land, and sea-ice components coupled through exchange of mass, momentum, energy, and chemical species. The components in the CCSM can be run standalone and are often used to examine specific science issues in runs that utilize high resolution spatial grids for relatively short simulation times and make use of thousands of processors. In climate scenarios utilizing the full CCSM the simulation times range from tens to thousands of years. As the time direction is not parallelized currently, throughput considerations limit the spatial resolutions that can be used in most production runs. This spatial resolution limitation itself limits the number of processors that can be exploited efficiently in the current parallel implementations based on spatial domain decomposition. Improvements in the parallel scalability of the CCSM components translate both to higher scientific throughput and to the ability to use (somewhat) larger spatial resolutions, thus increasing simulation fidelity. Improving parallel scalability in the CCSM is an important technique for accelerating climate science.

Within the Department of Energy (DOE) Scientific Discovery Through Advanced Computing (SciDAC) program, the project *A Scalable and Extensible Earth System Model for Climate Change Science* (SEESM) is in particular concerned with CCSM development, working to transform the CCSM into an earth system model that fully simulates the coupling between the physical, chemical, and biogeochemical processes in the climate system. Associated with SEESM is the Science Application Partnership project *Performance Engineering for the Next Generation Community Climate System Model* (PENG). PENG is tasked with the long-term

performance engineering of the CCSM, including taking into account the new physical processes that CCSM will include in the future. Note that this work is done in collaboration with CCSM Software Engineering Group [1] at the National Center for Atmospheric Research and with the CCSM Software Engineering Working Group [2].

Current PENG activities include porting CCSM to the IBM BlueGene/L system, optimizing CCSM performance on the Cray XT4, introducing parallel I/O into CCSM, analyzing and minimizing CCSM memory requirements, evaluating the performance of new numerical algorithms being proposed to replace those currently used, and analyzing and improving the scalability of CCSM components.

The sea ice and land components are surface models (essentially two-dimensional spatially), and have a lower computational complexity than the three-dimensional ocean and atmosphere models in typical CCSM runs. Moreover, the current parallel implementations of the land and sea ice components support two-dimensional domain decompositions and have been shown to scale reasonably well as a function of processor counts. Within the context of the CCSM, neither the sea ice nor the land represent a performance scalability bottleneck. The atmosphere and ocean models have roughly similar computational requirements, though the choice of spatial resolution or physical processes for each component can make one significantly more expensive than the other. As shown in Figure 1, the ocean component POP (Parallel Ocean Program) can use over 4000 processes for a small one-degree-resolution spatial grid, though parallel efficiency drops off when using more than 512 processes on the Cray XT4, and over 20,000 processes for a large eddy-resolving tenth-degree-resolution spatial grid. In contrast, the atmosphere component CAM (Community Atmosphere Model) has severe algorithmic limitations on the number of MPI processes that can be used. As shown in Figure 2, for the current production numerical algorithm and spatial resolution, at most 128 MPI processes can be used. For the next-generation numerical algorithm at a resolution larger than currently used in production, at most 960 MPI processes can be used. OpenMP parallelism can be used to push processor count scalability beyond these limits, but this is not available on all systems, nor is it efficient to use large numbers of OpenMP threads per process on most high performance computing systems.

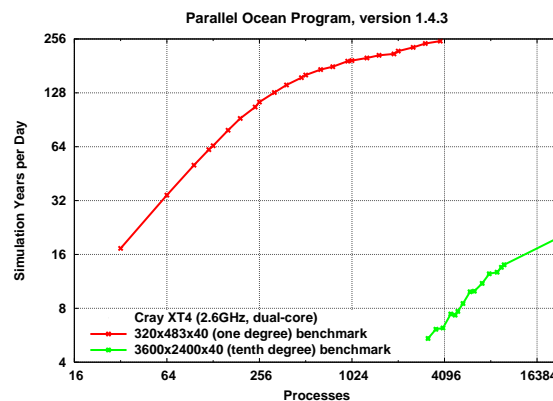


Figure 1. Processor-count scalability of the CCSM ocean component for small and large problem sizes.

In this paper, we describe current efforts in improving the scalability of CAM, the atmosphere component of CCSM.

2. Community Atmosphere Model

CAM is a global atmosphere model developed at the National Science Foundation's National Center for Atmospheric Research with contributions from researchers funded by DOE and by the

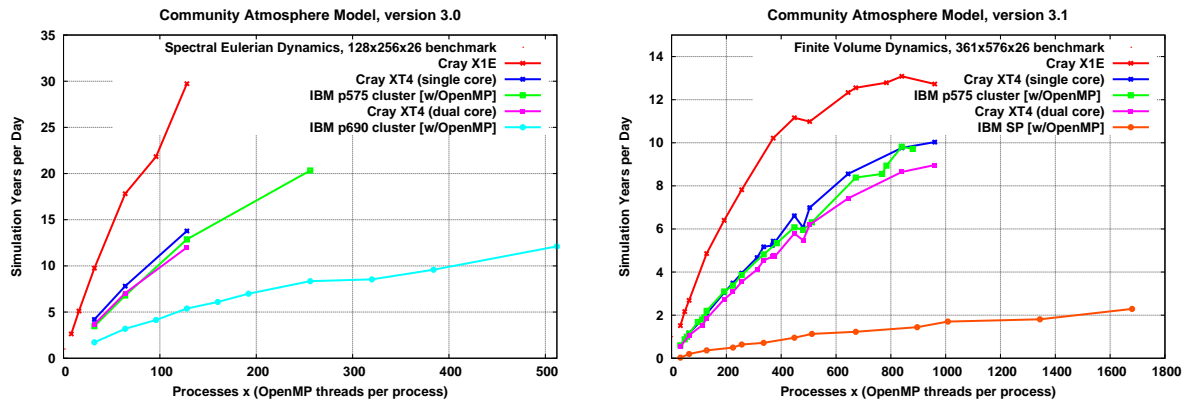


Figure 2. Processor-count scalability of the CCSM atmosphere component.

National Aeronautics and Space Administration [4]. CAM is a mixed-mode parallel application code, using both the Message Passing Interface (MPI) [9] and OpenMP protocols [8]. 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 [5]. 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 [12], a spectral semi-Lagrangian [15], and a finite-volume semi-Lagrangian [13]. 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 spectral Eulerian dycore had been the preferred dycore for CCSM production runs for the past several years. However, the finite-volume dycore has just recently become the preferred dycore, due to superior conservation properties that are important for a number of physical processes that are being introduced into the model. However, all of the dycores will be used for science studies for the foreseeable future.

The parallel implementations of the spectral dycores are based on a one-dimensional latitude decomposition. Thus, for the standard $128 \times 256 \times 26$ (latitude by longitude by vertical) grid, at most 128 MPI processes can be used. In contrast, the parallel implementation of the physics is based on an arbitrary latitude-longitude decomposition, and could use up to 32,768 processes algorithmically. While two-dimensional decompositions have been developed for the spectral dycores in the past, significant software re-engineering would be required, and this level of effort will not be expended on dycores that are otherwise no longer under development.

The parallel implementation of the finite-volume dycore is based on two different two-dimensional decompositions, one over latitude-longitude and one over latitude-vertical. In these decompositions, each subdomain must contain at least three grid points in each coordinate direction. The same total number of MPI processes must be used in each decomposition. For the initial production problem resolution of $96 \times 144 \times 26$, the maximum number of MPI processes that can be used in the latitude-vertical decomposition is 256. For the latitude-longitude decomposition the limit is 1,536, while for the physics decomposition the limit is 13,824. Thus the decomposition over the latitude-vertical severely limits the scalability for this

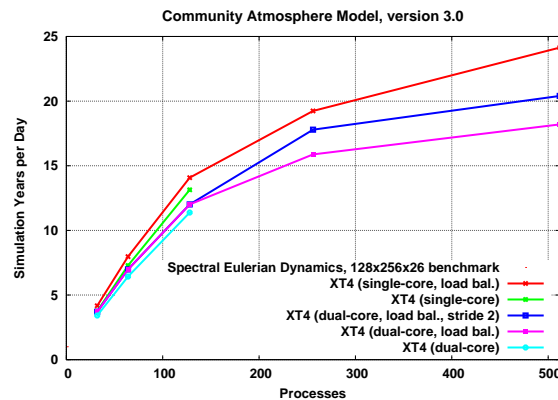


Figure 3. Performance impact of auxiliary MPI processes on spectral dynamics.

problem resolution.

3. Auxiliary MPI processes

As indicated in the previous section, the number of MPI processes that can be used varies between the dynamics and the physics, and between different phases of the dynamics for the finite-volume dycore. Our initial approach to improving the scalability of CAM has been to support idle, or *auxiliary*, processes. We are now able to allocate enough processes for the most parallel phases of the code, leaving some number of these idle during the less parallel phases.

Figure 3 shows the performance impact of auxiliary processes for the spectral Eulerian dycore on the Cray XT4. Physics load balancing is the mechanism used to redistribute work over a larger number of MPI processes in the physics than in the dynamics. Thus performance without physics load balancing suffers from both the original limitations on scalability and the performance impact of load imbalances. The XT4 uses dual-core Opteron processors, and using both processor cores (“dual-core”) incurs some performance degradation due to contention for memory and network access. Performance curves are presented for using only one core per processor (“single-core”), leaving the other idle, and using both cores. The curve labeled *dual-core, load bal., stride 2* uses every other processor core in the dynamics when using more the 128 processors in the physics. In this mode, the dynamics is running “single-core” while the physics is running “dual-core”, achieving performance between the pure “single-core” and “dual-core” modes without wasting as many computer resources as in the “single-core” mode.

The cost of the dynamics does not decrease when using more than 128 processes for this problem size. The cost of the physics decreases almost linearly out to 1024 processors. Because the physics is twice as expensive as the dynamics when using 128 processes, the physics and dynamics have comparable cost at 1024 processors, and there is little to be gained by increasing the processor count further. However, this modification increased throughput by a factor of 1.6

Tables 1–3 show the performance impact of auxiliary processes for the finite-volume dycore on the Cray XT4. Here the same number of MPI processes are used in the latitude-longitude decomposition and in the physics, and processes are idle only during the latitude-vertical decomposition phase. In all examples, the smallest processor count is the maximum for the latitude-vertical phase. While the cost of the physics shows the expected decrease as a function of processor count, the cost of the dynamics is increasing. We believe that this is a performance bug, and we will be addressing this in future work. In particular, the “stride-2” optimization used in the spectral dycores has not yet been implemented in the finite-volume dycore. The ability to use additional processors increased throughput by a factor of 1.2. In the finite-volume

processors	total	dynamics	physics	non-dynamics communication
256	3.63	1.37	1.61	0.26
512	2.94	1.61	0.83	0.35
1024	3.05	1.88	0.44	0.44

Table 1. Secs/day for phases when using finite-volume dycore with $96 \times 144 \times 26$ grid.

processors	total	dynamics	physics	non-dynamics communication
512	8.80	4.38	3.00	0.89
1024	6.95	4.54	1.60	0.38
2048	7.45	5.34	0.80	0.63

Table 2. Secs/day for phases when using finite-volume dycore with $192 \times 288 \times 26$ grid.

processors	total	dynamics	physics	non-dynamics communication
960	24.9	16.5	6.00	1.13
1920	21.0	15.9	3.10	0.85

Table 3. Secs/day for phases when using finite-volume dycore with $361 \times 576 \times 26$ grid.

dynamics, the dynamics is more expensive than the physics for all but the smallest spatial resolutions in the current version of the model, and there is less to be gained by exploiting additional parallelism in the physics.

4. Model evolution

There are a number of changes planned for CAM, all of which will increase the computational complexity. We discuss only two here.

First, the addition of atmospheric chemistry will double or triple the cost of the physics. Atmospheric chemistry will also require the advection of approximately 100 new tracers, increasing the cost of the dynamics as well. Figure 4 shows the increase in the runtime of CAM due solely to the increase in the number of advected tracers (not including the cost of atmospheric chemistry). This factor of 3.5 may yet be decreased with further optimization of tracer advection, but it also may be addressed by the use of auxiliary processes. For example, the advection takes place within the latitude-vertical decomposition phase, and different species can be advected independently, thus increasing the exploitable parallelism. Some of the advection can also be overlapped with the the rest of the phase.

Second, there is strong interest in resolving cloud processes within CAM. One approach is to reduce the horizontal grid spacing to 5 kilometers or less, thus increasing the grid size by a factor of 400 or more compared to current production resolutions, and reworking all of the subgrid parameterizations. An alternative, referred to as *superparameterization*, involves increasing the resolution within each cell of the standard resolution for the cloud processes only [11]. This increases the computational cost of each grid point in the original grid by a factor of 100, which makes the use of auxiliary processes in the physics an important performance enhancement. An experiment with such an approach using the spectral semi-Lagrangian dycore on a $64 \times 128 \times 26$ grid demonstrated that 1024 processes could be used effectively. In fact, this pure MPI implementation was more efficient than utilizing 64 MPI processes and 16 OpenMP threads per process on a cluster of IBM SMP nodes [10].

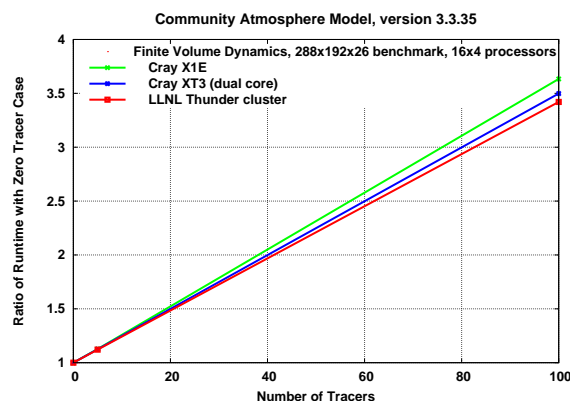


Figure 4. Performance impact of tracer advection on finite-volume dycore

5. Conclusions

This paper briefly describes current efforts to improve the scalability of the Community Atmosphere Model by adding support for auxiliary MPI processes that are idle during some phases of the code and active in others. While this does not change the asymptotic scalability, it has been shown to increase throughput in the current version of CAM and will be even more important as new physics is introduced into the model. We are still actively optimizing the current implementation. We are also investigating alternative numerical methods with improved scalability compared to the current algorithms. However, support for auxiliary MPI processes is a technique that we expect to be useful even for more scalable dycores that are currently under development [7, 14].

Acknowledgments

This research used resources (Cray X1E, Cray XT4, 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 SP, 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 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-PROC-232512. The work of 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-Battelle, LLC. 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] CCSM SOFTWARE ENGINEERING GROUP. <http://www.cesm.ucar.edu/cseg/>.
- [2] CCSM SOFTWARE ENGINEERING WORKING GROUP. http://www.cesm.ucar.edu/working_groups/Software/.
- [3] W. D. COLLINS, C. M. BITZ, M. L. BLACKMON, G. B. BONAN, C. S. BRETHERTON, J. A. CARTON, P. CHANG, S. C. DONEY, J. H. HACK, T. B. HENDERSON, J. T. KIEHL, W. G. LARGE, D. S. MCKENNA, B. D. SANTER, AND R. D. SMITH, *The Community Climate System Model Version 3 (CCSM3)*, J. Climate, 19 (2006), pp. 2122–2143.

- [4] 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).
- [5] W. D. COLLINS, P. J. RASCH, 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.
- [6] COMMUNITY CLIMATE SYSTEM MODEL. <http://www.cesm.ucar.edu/>.
- [7] CUBED-SPHERE FINITE-VOLUME DYNAMICAL CORE. <http://sivo.gsfc.nasa.gov/cubedsphere.html>.
- [8] L. DAGUM AND R. MENON, *OpenMP: an industry-standard API for shared-memory programming*, IEEE Computational Science & Engineering, 5 (1998), pp. 46–55.
- [9] W. GROPP, M. SNIR, B. NITZBERG, AND E. LUSK, *MPI: The Complete Reference*, MIT Press, Boston, 1998. second edition.
- [10] M. KHAIROUTDINOV, April 2007. personal communication.
- [11] M. KHAIROUTDINOV, D. RANDALL, AND C. DEMOTT, *Simulations of the atmospheric general circulation using a cloud-resolving model as a superparameterization of physical processes*, Journal of Atmospheric Sciences, 62 (2005), pp. 2136–2154.
- [12] 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.
- [13] S.-J. LIN, *A “vertically Lagrangian” finite-volume dynamical core for global models*, Mon. Wea. Rev., 132 (2004), pp. 2293–2307.
- [14] R. D. LOFT, S. J. THOMAS, AND J. M. DENNIS, *Terascale spectral element dynamical core for atmospheric general circulation models*, in Proceedings of the ACM/IEEE Conference on High Performance Networking and Computing (SC01), Nov. 10-16, 2001, Los Alamitos, CA, 2001, IEEE Computer Society Press. <http://www.homme.ucar.edu>.
- [15] 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.