

International Journal of High Performance Computing Applications

<http://hpc.sagepub.com/>

Improved parallel performance of the CICE model in CESM1

Anthony P Craig, Sheri A Mickelson, Elizabeth C Hunke and David A Bailey

International Journal of High Performance Computing Applications published online 3 September 2014

DOI: 10.1177/1094342014548771

The online version of this article can be found at:

<http://hpc.sagepub.com/content/early/2014/09/03/1094342014548771>

Published by:



<http://www.sagepublications.com>

Additional services and information for *International Journal of High Performance Computing Applications* can be found at:

Email Alerts: <http://hpc.sagepub.com/cgi/alerts>

Subscriptions: <http://hpc.sagepub.com/subscriptions>

Reprints: <http://www.sagepub.com/journalsReprints.nav>

Permissions: <http://www.sagepub.com/journalsPermissions.nav>

Citations: <http://hpc.sagepub.com/content/early/2014/09/03/1094342014548771.refs.html>

>> [OnlineFirst Version of Record](#) - Sep 3, 2014

[What is This?](#)

Improved parallel performance of the CICE model in CESM¹

Anthony P Craig¹, Sheri A Mickelson², Elizabeth C Hunke³
and David A Bailey¹

*The International Journal of High
Performance Computing Applications*
1–12

© The Author(s) 2014

Reprints and permissions:

sagepub.co.uk/journalsPermissions.nav

DOI: 10.1177/1094342014548771

hpc.sagepub.com



Abstract

The Los Alamos sea ice model, CICE, is a sophisticated finite difference grid point model. It has been a part of the National Science Foundation and Department of Energy community climate models (Community Climate System Model (CCSM) and Community Earth System Model (CESM)) for over a decade. It includes various physical and dynamical processes and is parallelized to run on large-scale computer systems. The CICE model was assessed in the CESM at different resolutions and target processor counts to better understand and optimize the performance. Several new decompositions and a new feature to reduce the halo cost were added to the model. The new decompositions better leverage land block elimination and take advantage of scaling opportunities in different computational kernels. As a result of these new features, the CICE model performance has been improved by up to 45% and has more flexibility to be run efficiently at arbitrary Message Passing Interface (MPI) task counts.

Keywords

Climate model, performance, CICE, decomposition, optimization

1. Introduction

The CICE model (Hunke et al., 2013) is a sophisticated finite difference grid point model that is developed and maintained at the Department of Energy's (DOE's) Los Alamos National Laboratory (LANL). It consists of various physical and dynamical processes, including multiple category sea ice thermodynamics (Bitz and Lipscomb, 1999), delta-Eddington radiation (Briegleb and Light, 2007), horizontal transport via incremental remapping (Dukowicz and Baumgardner, 2000; Lipscomb and Hunke, 2004), and elastic-viscous-plastic (EVP) dynamics (Hunke and Dukowicz, 1997, 2002). It has been modified for use in the Community Earth System Model (CESM; Gent et al., 2011; www2.cesm.ucar.edu), a freely available community model developed by researchers funded by the US DOE, the National Aeronautics and Space Administration (NASA), and the National Science Foundation (NSF), and maintained at the National Center for Atmospheric Research (NCAR). The physical processes modeled in the CESM version are largely identical to those in the CICE v4.1 version based at LANL.

Prior studies (Zhang and Hibler, 1997; Losch et al., 2010; Lemiux et al., 2012) evaluated the scientific and computation performance of sea ice models as a function of solver, dynamics, physics, or time-stepping schemes. Those results demonstrated the sensitivity of the model solution to the algorithms used. However,

few detailed studies have been published on the basic computational cost and scaling of a sea ice model in order to primarily improve efficiency, cost, and throughput. This study assesses the detailed performance of the CICE model at a fixed configuration by varying processor counts and testing various decomposition strategies. The goal is to better understand the CICE model performance and then to be able to leverage that information in the future to run the model efficiently and improve CICE computational efficiency on high-performance computing hardware.

The CICE model grid is based on a two-dimensional orthogonal horizontal grid. The model grid in the CESM is typically either a displaced pole or tripole global grid with properties similar to a longitude/latitude grid. The dipole grid shifts the North Pole to a region over land that stretches the grid in the Northern

¹Climate and Global Dynamics Division, National Center for Atmospheric Research, USA

²Mathematics and Computer Science Division, Argonne National Laboratory, USA

³Fluid Dynamics and Solid Mechanics Group, Theoretical Division, Los Alamos National Laboratory, USA

Corresponding author:

Anthony Craig, c/o Sheri Mickelson, NCAR, PO Box 3000, Boulder, CO 80307, USA.

Email: tcraig@ucar.edu

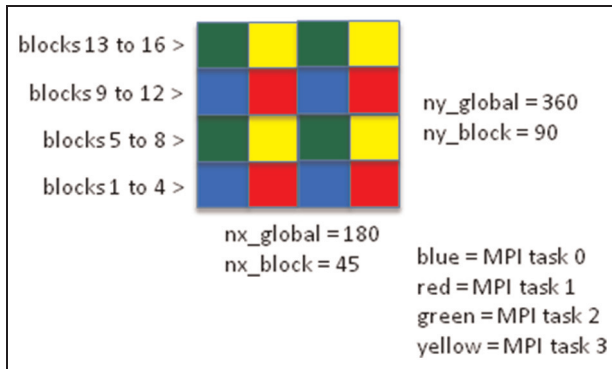


Figure 1. Example of a typical CICE model decomposition for a global grid of size 36×180 , decomposed into 16 blocks in a 4×4 fashion. Each block is 90×45 gridpoints and the blocks are then distributed to four different MPI tasks denoted by block color.

Hemisphere and avoids the polar singularity. The tri-pole grid has three singularities over land with two singularities in the Northern Hemisphere connected by a set of neighboring gridcells known as the “zipper”. Regional stereographic grids can also be configured and run with the CICE model. The CICE model contains multiple ice thickness categories per grid cell and computes thermodynamic variables in the vertical direction, but the vertical dimension is not used for parallelization. The horizontal grid in CICE is decomposed into two-dimensional blocks for parallelization. The blocks are distributed to MPI tasks, and each task can be assigned multiple blocks. While the size of the local blocks is constant across the grid, the number of blocks per task can vary. Threading, when activated in CICE, is also over blocks on each MPI task. Figure 1 provides a schematic of a CICE grid decomposition that has a global horizontal grid size of 360×180 grid cells and is decomposed into sixteen 90×45 sized blocks that are distributed to four MPI tasks. The user controls the decomposition in CICE with several different parameters. The block size for each horizontal direction (nx_block and ny_block in Figure 1) and the maximum number of blocks per task ($maxblocks$) are compile-time settings to minimize dynamic memory allocation, which can degrade model performance. The decomposition strategy that describes how to distribute the blocks to tasks is a run-time setting. CICE allows block sizes that do not decompose the grid exactly as well as the capability for MPI tasks to own zero blocks. In some cases, the decomposition of the ice grid results in blocks that contain all “land” grid cells. These blocks are not allocated to any MPI task. Because these blocks are known at initialization, decompositions can leverage that information to distribute active blocks more evenly among tasks. The advantages of land block elimination as part of the overall decomposition strategy are summarized well by Dennis (2007).

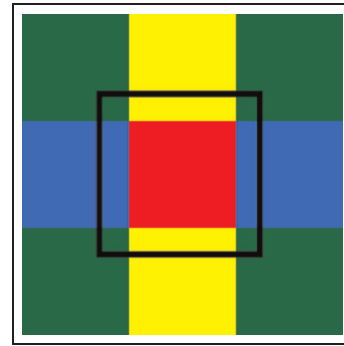


Figure 2. Schematic of a block and the halo associated with the block. The black line encloses the halo region associated with the center block.

The primary MPI communication cost in CICE is the halo update. CICE has some global reductions but only for diagnostics, and these are typically minimized in production runs. The halo update communicates local neighbor information between blocks in order to support finite difference calculations. The CICE algorithms used in the CESM require a single layer of “ghost cells” for each CICE block. Every CICE block has eight unique neighbors, one each to the east, northeast, north, northwest, west, southwest, south, and southeast. The northeast, northwest, southwest, and southeast are corner neighbors that share a single grid cell in the halo, while the east, north, west, and south neighbors share an edge of grid cells. Figure 2 shows schematically how the halo for a single block is associated with eight neighbor blocks. Halo communication in CICE is aggregated between pairs of MPI tasks. In other words, in cases where multiple neighbor blocks share the same MPI task, communication of data between MPI tasks over multiple blocks is done with a single MPI message. In addition, halo updates for data between blocks that are on the same MPI task are done using a local copy instead of messaging through the MPI interface.

The CICE model has become relatively more expensive in the CESM in the last few years for several reasons. New scientific algorithms that are more expensive have been added, and while CICE is not typically the most expensive model in the CESM, other models have improved their performance and scaling capabilities in the last few years. As a result, an effort was made to assess and improve CICE performance in the CESM, and this paper represents the results of that work. Section 2 provides an analysis of the CICE performance and summarizes work that was done in CICE to improve performance. Section 3 presents performance results, and Section 4 summarizes this work.

2. CICE cost assessment

Figure 3 is a stacked bar chart showing the CICE timing on 16, 64, 320, and 1280 MPI tasks with threading

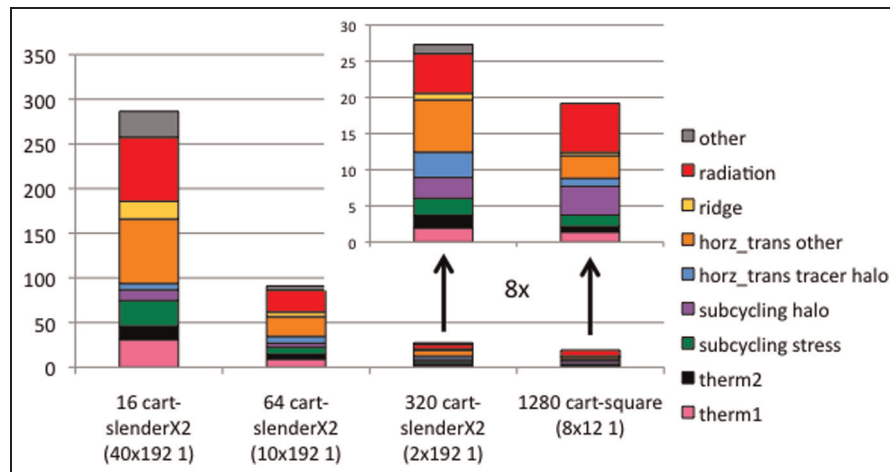


Figure 3. CICE performance for $g \times l$ (320 × 384) global configuration on hopper as a function of MPI tasks for task counts of 16, 64, 320, and 1280 broken down by computation kernel. The timed kernels are therm1 (pink), therm2 (black), subcycling stress (green), subcycling halo (purple), horizontal transport tracer halo (blue), horizontal transport tracer non-halo (orange), ridging (yellow), radiation (red), and other (gray). The vertical axis is seconds for a 10-day run. The 320 and 1280 results are also blown up by 8× for clarity. (Color online only.)

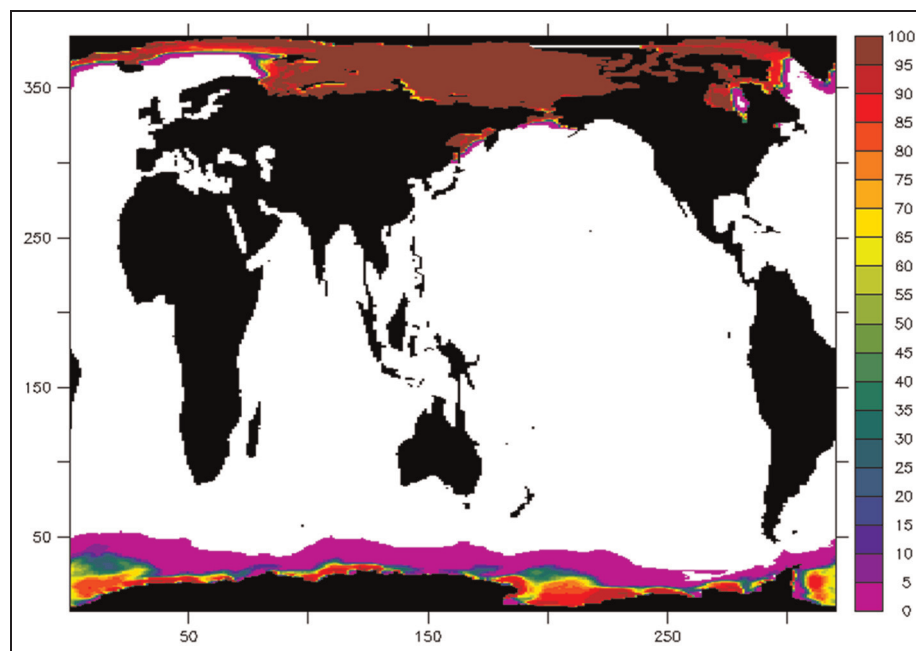


Figure 4. CICE $g \times l$ (320 × 384) displaced pole grid and sample ice concentration in percent. The displaced pole lies in Greenland, which is stretched across the top of this logical-space plot.

turned off on hopper, a CRAY XE6 at the National Energy Research Scientific Computing Center (NERSC) with 2.1 GHz Opteron processors and a Gemini interconnect, for a 320 × 384 displaced pole $g \times l$ “one-degree” configuration using some typical CICE block sizes and decompositions. In general, threading is not explored in this paper because blocks are used both in MPI parallelism and threading parallelism, so the fundamental parallelism in CICE does not change with threading. While the performance might

be slightly different, the relevant block sizes and decompositions associated with a problem run on 320 MPI tasks unthreaded and 80 MPI tasks with each task threaded four ways are largely identical. All CICE timings in this paper were done within the coupled CESM system using 10-day runs in January, and the CICE time was isolated from the rest of the system with MPI barriers. The $g \times l$ grid is shown in Figure 4. The $g \times l$ “one-degree” grid is the most widely used ice resolution in the CESM by far. Results for the high-

resolution $t \times 0.1$ “one-tenth degree” grid will also be presented later. The overall scaling in Figure 3 is largely as expected with 280, 90, 27, and 19 seconds per 10-day run for 16, 64, 320, and 1280 MPI tasks, respectively.

The CICE model consists of several different computational kernels and no single kernel dominates the cost. However, as shown in Figure 3, most of the cost of the CICE model occurs in a handful of kernels. These kernels are the *therm1* and *therm2* thermodynamic routines, the *ridging* subroutine, the radiation module, the *EVP* subcycling, and the horizontal transport. The *therm1* routine executes the primary vertical thermodynamic calculations, while *therm2* redistributes ice in the subgrid-scale thickness categories and forms new ice. Similarly, the *ridging* subroutine computes the deformation of the sea ice under stress and strain, leading to a change in the subgrid-scale thickness distribution. The radiation code is purely a vertical calculation and uses a multiple scattering algorithm to determine the reflected, absorbed, and penetrating solar radiation in sea ice. The *therm1*, *therm2*, *ridging*, and radiation costs are dominated by computations with little or no communication required in these methods. The *EVP* dynamics code solves a set of explicit equations for the sea ice velocities and stresses. Additional damping is provided in the *EVP* implementation via subcycling. One-hundred-and-twenty subcycles are typically used in standard configurations. The subcycling requires communication with neighboring grid cells, so halo updates are required at every subcycle. The horizontal transport of sea ice properties, such as the fraction, thickness, and energy, also requires communication with neighboring grid cells. These are carried out as large multi-field multi-level halo updates. To better understand the performance scaling in CICE, the halo portion of the *EVP* subcycling and horizontal transport is timed separately from their computational cores.

In Figure 3 at 16 tasks, the delta-Eddington radiation and the non-halo part of the horizontal transport are the two most expensive kernels with respect to cost. At 320 tasks, the halo kernels (horizontal transport tracer halo and *EVP* subcycling halo) take up a greater percentage of the total time as expected. In all cases, the total time and the kernel time are computed via two distinct runs. Kernel times are generated by adding MPI barriers before each kernel to isolate load imbalances. CICE performance can be attributed to two primary aspects; firstly, the computational cost on each MPI task and how well that work is load-balanced between tasks and, secondly, the communication (halo) cost.

While the computational work in CICE is trivially parallel, the amount of work on each task varies in time depending on the sea ice distribution. CICE computations occur only where sea ice exists in the model, and the sea ice distribution continuously varies in time,

particularly at seasonal timescales. For global grids in the CESM, most of the grid is ice-free all year and many grid cells are ice-free part of the year, as suggested by Figure 4. In addition, the radiation kernel, which is relatively expensive in CICE, is only computed where sea ice exists and where the sun is up at any fixed time. This solar dependence creates a large diurnal and seasonal influence on the cost of the radiation kernel. Thus, the cost of the CICE model depends heavily on the amount of the grid covered by sea ice at any time, how that distribution varies in time, and how the grid is decomposed. The model performs only as well as the slowest task. To optimize load balance, each task should have about the same number of ice-occupied grid cells at any given time. In practice, this is not possible with a static decomposition with temporally varying sea ice coverage, but the grid decomposition strategy has a big influence on the model performance. Dynamic load balance algorithms are being considered for the CICE model and could be attractive because the individual gridcell cost varies primarily on seasonal timescales rather than faster timescales.

The cost of the halo communication is a function of both the number of messages sent and the amount of data sent. The number of unique MPI tasks associated with each block's neighbor blocks determines the number of messages needed for each task, and neighbor blocks that are assigned to the same task greatly reduces the cost of the halo communication by avoiding MPI completely. Large blocks and low aspect ratio blocks minimize the amount of data that needs to be sent. As a result, the halo cost is influenced by both the size and distribution of the CICE blocks.

Unfortunately, the optimal block distribution for best load balance is nearly orthogonal to the optimal block distribution for minimum halo cost. The model is often best load-balanced when blocks from multiple regions of the grid (i.e. high and low latitude, northern and southern hemisphere, eastern and western hemisphere) are assigned to each MPI task, but this tends to degrade the communication performance. On the other hand, the communication cost is minimized when either one block per task or multiple collocated blocks per task are assigned to an MPI task, but this often results in poor computational load balance.

As a result of this study, two different modifications were made to the CICE model to improve performance. The halo update was modified to eliminate some unnecessary communication, and new decompositions were added that provide alternative options for better balancing computational and communication cost.

2.1. Masked halos

The halo communication routing in CICE is computed at initialization based on the static block

decomposition. The halo update assumes all halo data is needed, so communication patterns are established at initialization that communicate all halo data across the grid. However, in practice, at any given time, many grid cells have no sea ice. In that case, “zeros” are sent via the halo update with no actual benefit to the solution. A new feature was introduced in CICE where the halo update can be modified at any time to exclude communication of halo information in areas of the grid where it is not needed. In order to compute the masked halo associated with sea ice concentration, for instance, a full halo update is first done on the sea ice concentration field. That information is then converted to a mask and used to compute a new masked halo via a purely local computation. That masked halo can then be applied to fields as long as the sea ice concentration is static. As a result of the upfront cost (primarily an extra halo call), this optimization is not beneficial unless the masked halo can be subsequently applied to multiple halo calls. There are several subroutines where the masked halo can be applied effectively in CICE. A masked halo can be initialized at the start of the EVP subcycling loop and reused throughout the subcycling, the multiple halo updates in the EVP stress computation for tripole grids can be done with a single masked halo, and the multiple multi-field halo updates in subroutine `bound_state` and subroutine `horizontal_remap` benefit from a masked halo. Masked halos are controlled by new CICE namelist variables, and are “on” by default in CESM1. The masked halos improve performance by as much as 25% across a wide range of cases by reducing the amount of data that is communicated in the halo update in CICE.

2.2 CICE decompositions

There are three block distribution types (or strategies) in CICE v4.1; Cartesian, rake, and spacecurve. The Cartesian and rake distribution types are further refined into processor shapes `slenderX1`, `slenderX2`, `square-ice`, and `square-pop`. Three additional distribution types were added in CESM1.1 and in CICE v5: `roundrobin`, `sectrobin`, and `sectcart`. This study focuses on seven of the decomposition strategies; `cartesian-slenderX1`, `cartesian-slenderX2`, `cartesian-square` (`square-pop`), `sectcart`, `spacecurve`, `roundrobin`, and `sectrobin`. These decompositions were chosen because they have been used historically and/or tend to perform the best. Figure 5 shows the block distribution of these seven decomposition strategies when applied to a global horizontal $320 (i)$ by $384 (j)$ $g \times 1$ grid decomposed into 256 blocks of size 20×24 and distributed to 16 MPI tasks. The color scale in Figure 5 represents the MPI task assigned to each block.

The `cartesian-slenderX1` decomposition is shown in Figure 5(a). Blocks are allocated to tasks such that each task has a set of blocks that span the full extent of the

grid in the j direction. The `cartesian-slenderX1` is a one-dimensional decomposition in the i direction with respect to tasks and can only be run on MPI task counts that divide the i grid size evenly. This decomposition has several benefits. Firstly, the number of neighbors is fixed at two and the load balance is generally good because each task spans the full j direction, which for global grids is often associated with the latitudinal direction. The downside is that the size of the halo is quite large because the aspect ratio of the blocks tends to be high. This reduces the efficiency of the halo update by increasing the amount of data that needs to be sent.

The `cartesian-slenderX2` decomposition is shown in Figure 5(b). This decomposition is identical to the `cartesian-slenderX1` decomposition except the blocks for a given task span half the extent in the j direction. This has similar performance benefits and constraints compared to the `cartesian-slenderX1`, but the halo size is smaller because the aspect ratio of the blocks is lower for a given MPI task count. It can be run on twice as many MPI tasks as the `cartesian-slenderX1` decomposition, and each block only covers half the extent of the j direction.

The `cartesian-square` (`square-pop` in CICE v4.1) decomposition is shown in Figure 5(c). It is a general two-dimensional decomposition that is identical to the `cartesian-square` decomposition in the POP ocean model. In this case, neighboring blocks are allocated to a given MPI task, and the decomposition requires that a block size be chosen that allows an equal number of blocks be given to each MPI task in both the i and j directions. While this limits the block sizes that can be chosen, this decomposition has several good properties including the fact that several neighbor blocks often share the same MPI task, the maximum number of MPI neighbors will never be greater than eight, and the size of the halo is minimized for a given decomposition. All of these properties produce efficient communication performance. The trade-off is that this decomposition results in blocks that are closely collocated physically, which generally leads to poor load balance of sea ice work across tasks on a global CESM grid.

The `sectcart` decomposition is shown in Figure 5(d) (some release versions of the CESM call this `blkcart` instead of `sectcart`). In this decomposition, the grid is divided into equal quarters and the blocks in each quadrant are assigned to all the MPI tasks using a simple one-dimensional decomposition. The blocks are distributed starting from the bottom left of each quadrant for the two left quadrants and from the top left of each quadrant for the two right quadrants. This distribution scheme means each MPI task has the same number of blocks from each quadrant, each task has the same neighbor tasks in each quadrant, and each MPI task has blocks from different regions of the grid. In many

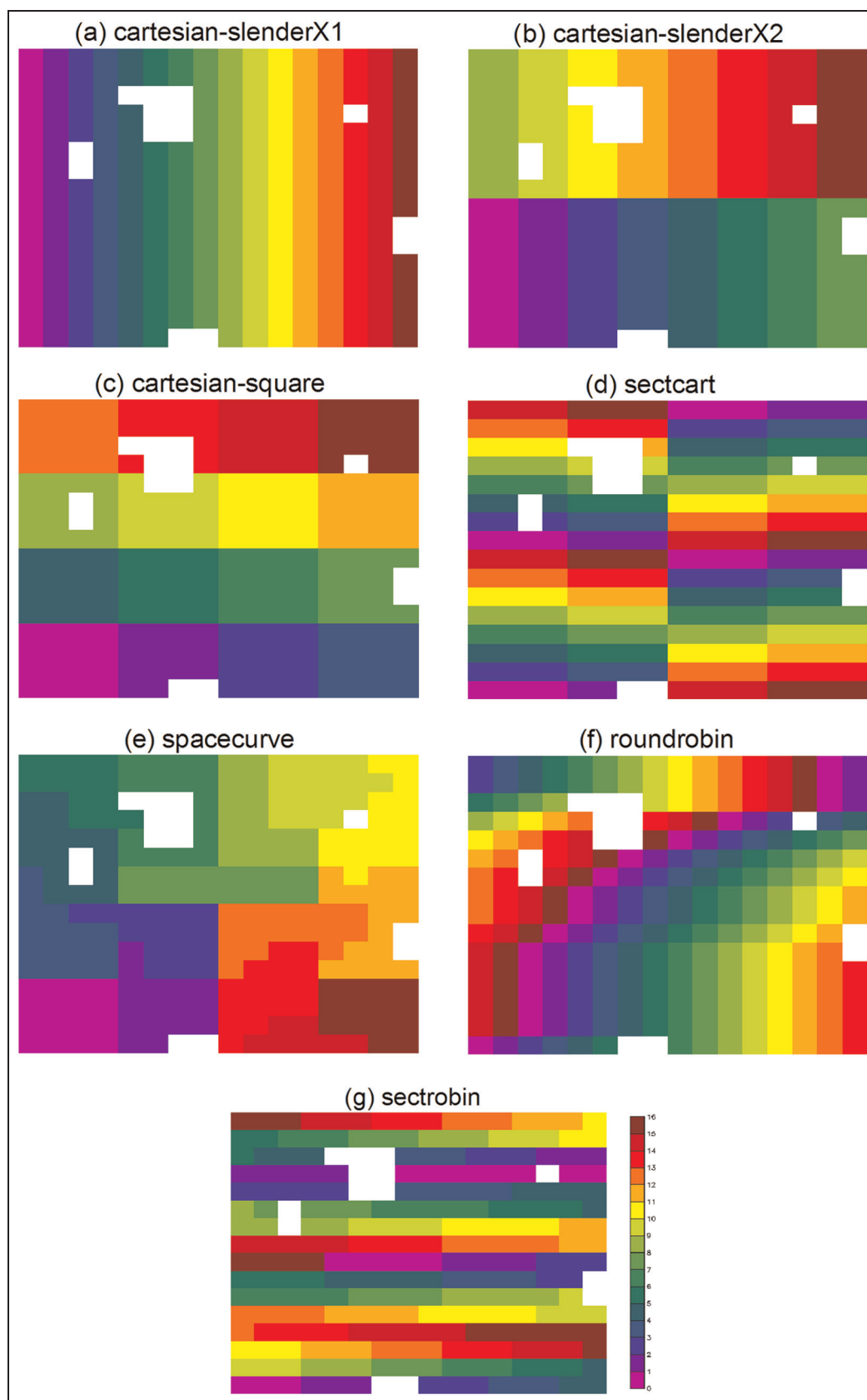


Figure 5. CICE decomposition options in CESM1: (a) cartesian-slenderX1; (b) cartesian-slenderX2; (c) cartesian-square; (d) spacecurve; (e) sectcart; (f) roundrobin; (g) sectrobin for a global 384×320 “g×l l” grid, decomposed into 20×24 sized blocks and distributed onto 16 MPI tasks. Colors indicate the MPI task to which each block is assigned. Blocks that contain only land gridcells (white) are not assigned to an MPI task. (Color online only.)

ways, this decomposition is like the cartesian-square decomposition applied to each quadrant. It has reduced communication cost like the cartesian-square decomposition, while better distributing blocks from different regions of the grid to each MPI task. This decomposition requires that there are at least four blocks per task.

The spacecurve decomposition (Dennis, 2007) is shown in Figure 5(e). This decomposition supports more flexible MPI task counts and leverages land block elimination. In this decomposition, the block size must be defined such that the number of blocks in the i and j directions are divisible only by 2, 3, and 5. The blocks are then distributed to tasks along a Hilbert–m–Peano–Cinco spacecurve on the blocked grid. The spacecurve decomposition has many nice features. The size of the halo is generally relatively low and the number of neighbors is also relatively low. The number of MPI tasks that can be chosen is very flexible and the land-only blocks are excluded upfront, leading to more efficient distributions. However, because the blocks tend to be physically collocated like the cartesian-square decomposition, the overall load balance is often not optimal. Some improvements are being pursued within the spacecurve decomposition to take into account the cost of blocks in order to better load balance the decomposition, but that feature is not yet generally available.

The roundrobin decomposition is shown in Figure 5(f). It was added to provide a flexible decomposition when the CICE model is run in prescribed mode, dynamics are turned off, and there is no communication cost. However, it can be applied just as easily when the CICE model is run in fully prognostic mode. The roundrobin decomposition distributes blocks to tasks in a purely roundrobin fashion after removing the land-only blocks. The distribution is done with the i direction running fastest. This decomposition can be applied to any block sizes and any task counts. If there are multiple blocks per task, this decomposition tends to result in well load-balanced decompositions because each task has blocks from multiple locations on the grid. However, the halo communication cost tends to

be very high because the number of neighbor tasks is relatively unconstrained, such that both the number of messages and amount of communicated data tends to be very large.

The sectrobin decomposition is shown in Figure 5(g) (some release versions of the CESM call this blkrobin instead of sectrobin). It attempts to leverage the positive feature of the well load-balanced roundrobin decomposition while reducing the communication cost of that decomposition. The sectrobin decomposition operates like the roundrobin decomposition, except it groups physically collocated blocks from different locations on the grid to an MPI task where possible, while still giving each MPI task blocks from different regions of the grid. Based on empirical testing, the distribution algorithm works best on CESM global grids when there are at least five blocks per task, and in those cases, it often provides a good compromise between load balancing the model cost across tasks and reducing the communication cost between tasks.

Table 1 provides a summary of the seven CICE decompositions analyzed in this study and a subjective characterization of some of their features based on their inherent properties and validated by the authors' experience. Each feature in each decomposition is subjectively rated poor, fair, good, or excellent based upon how well that feature is handled by each decomposition. Ideally, a decomposition would exist with excellent ratings in each category. Instead, the optimal decomposition strategy for any given configuration depends on hardware and target MPI task count. At lower task counts, load balance is most important. As task counts increase, MPI performance generally becomes more important. Determining the optimal CICE decompositions is complex, and a practical guide is available on the web at <http://oceans11.lanl.gov/drupal/CICE/DecompositionGuide>.

3. Performance results

Figure 6 is a stacked bar chart showing the timing of the CICE model in the CESM. In this case, multiple decompositions were analyzed for a fixed problem,

Table 1. A summary of the decompositions available in CICE in CESM1 as well as a subjective rating (poor, fair, good, or excellent) of several performance features.

Decomposition	Load balance In the “j” direction	Load balance in the “i” direction	Number of MPI messages	Amount of MPI data	Land block elimination enabled	Flexible task counts
cartesian-square-pop	Poor	Poor	Good	Excellent	Poor	Fair
cartesian-slenderX1	Excellent	Fair	Excellent	Fair	Poor	Poor
cartesian-slenderX2	Good	Fair	Good	Good	Poor	Poor
spacecurve	Poor	Poor	Good	Good	Excellent	Excellent
roundrobin	Excellent	Excellent	Poor	Poor	Excellent	Excellent
sectrobin	Good	Excellent	Fair	Fair	Excellent	Excellent
sectcart	Good	Excellent	Good	Fair	Poor	Fair

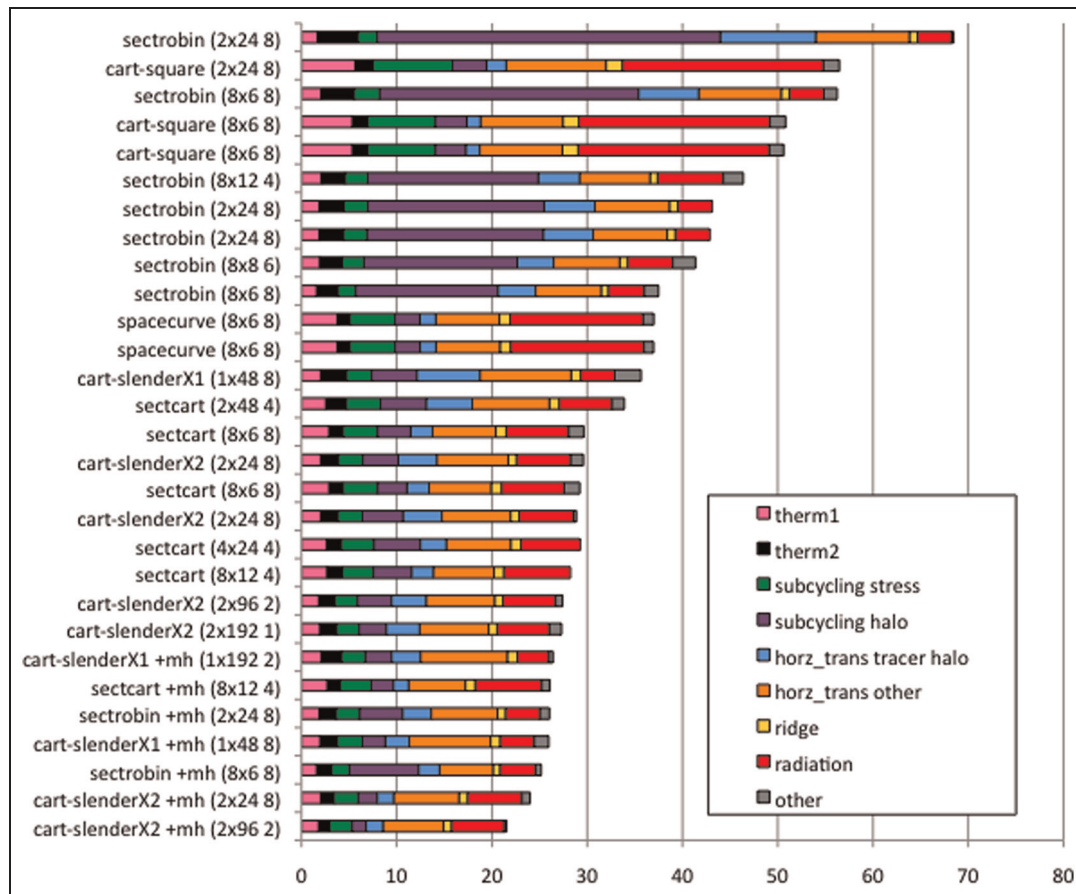


Figure 6. CICE performance versus decomposition for a $320 \times 384 \text{ g} \times 1$ grid on 320 MPI tasks on hopper broken down by computation kernel. The y-axis indicates the decomposition by name. Local block size and maximum blocks are specified in parentheses. Measurements with masked halos are indicated by +mh in the decomposition name. The x-axis is time in seconds for a 10-day run. The timed kernels are therm1 (pink), therm2 (black), subcycling stress (green), subcycling halo (purple), horizontal transport tracer halo (blue), horizontal transport tracer non-halo (orange), ridging (yellow), radiation (red), and other (gray). (Color online only.)

hardware, and task count, for the global $320 \times 384 \text{ g} \times 1$ grid on hopper on 320 MPI tasks with OpenMP threading turned off. Again, the $\text{g} \times 1$ grid is shown in Figure 4. Run lengths were 10 days, barriers were used to isolate kernel timing, and the time for each kernel represents the aggregate time over multiple calls for the slowest MPI task. Several cases were run multiple times to demonstrate performance reproducibility and one of these cases is presented in the chart. The decomposition (strategy, block size, and maxblocks) is presented along the y-axis and time is on the x-axis. The plot is sorted based on total time. The masked halo feature was turned on for a few decompositions for comparison. The decompositions chosen are relatively optimal and represent just a subset of decompositions tested.

Ignoring the masked halos for a moment, the total time of the CICE model in Figure 6 varies from 27 s for the cartesian-slenderX2 ($2 \times 96 \text{ 2}$) configuration to 68 s for the roundrobin ($2 \times 24 \text{ 8}$) configuration, indicating

the huge impact the decomposition can have on overall performance. Remember that in this case, CICE is being run on the same problem, same machine, and same MPI task count in all cases with bit-for-bit identical results. The relative performance of the load balance part of the code and the communication part of the code can also be easily seen in Figure 6. Comparing the size of the radiation timing (red) and the subcycling halo term (purple) in Figure 6 provides a sense of the relative efficiency of load balance versus communication for individual decompositions. The load balance driven kernels like radiation perform quite poorly in the cartesian-square and spacecurve decompositions and quite well in the sectrobin and roundrobin decompositions. The communication driven kernels like the subcycling halo are quite expensive for the roundrobin and sectrobin decomposition, but quite cheap in the cartesian-square and spacecurve decompositions. The cartesian-slender and sectcart decompositions are relatively well balanced with respect to computation and

communication cost, and they tend to be the best performing decompositions in this case.

One interesting pair of results to compare is the spacecurve ($8 \times 6 \times 8$) and the sectrobin ($8 \times 6 \times 8$). Their total times (about 37s) are nearly identical, but the relative cost of the various kernels varies significantly. The spacecurve decomposition performs better on the communication-dominated kernels and the sectrobin decomposition performs better on the load balance-dominated kernels. Individual kernels are as much as five times more expensive in one decomposition compared to the other decomposition for this pair of timings.

The masked halo feature improved the CICE performance for the cases tested in Figure 6 and is particularly effective for the sectrobin decomposition. Without the masked halo feature, the sectrobin decomposition performs average, at best, in Figure 6 with most of the cost associated with the subcycling halo communication. With the masked halo turned on, the communication costs are decreased significantly and the net result is that the sectrobin decomposition with masked halos becomes one of the best performing decompositions overall. Comparing the sectrobin ($8 \times 6 \times 8$) timing with masked halos and without, the subcycling halo cost improves by a factor of 2 and the total time decreases by about 25%. The amount of benefit of the masked halo feature depends on the underlying efficiency of the halo communication for the decomposition chosen.

Table 2 shows the sensitivity of the CICE performance to block size and number on a fixed problem and decomposition for a $320 \times 384 \times 1$ case on hopper on 320 MPI tasks using the cartesian-slenderX2 decomposition. The cartesian-slenderX2 is somewhat constrained in the number of tasks and block sizes that can be used, but it was chosen because it preserves the distribution of gridcells on tasks as blocks are varied. In this case, the best performance is with two blocks per task with little performance loss at one and four blocks per task. The times increase slowly from 2 to 16 blocks per task, and there is a large increase in time at 32 blocks per task. Blocks with less than 16 grid cells tended to underperform consistently throughout this study, as did configurations with greater than 16 blocks per task for all decompositions. For any decomposition strategy, there is an optimal block size. It is important to consider both the block size and the decomposition strategy when determining the optimal decomposition for any given CICE configuration.

Prior to this study, the CICE model ran efficiently on only a handful of different task counts for a given resolution by leveraging the cartesian-slenderX1 and cartesian-slenderX2 decompositions on the CESM global grids. At high resolutions and high task counts, some spacecurve decompositions were used, but a comprehensive performance assessment was missing.

Table 2. CICE performance versus blocksize for a $320 \times 384 \times 1$ grid on 320 MPI tasks on hopper using the cartesian-slenderX2 decomposition for a 10-day run.

Blocksize, blocks per task	Time (seconds)
$2 \times 192, 1$	27.6
$2 \times 96, 2$	27.0
$2 \times 48, 4$	27.5
$2 \times 24, 8$	28.7
$2 \times 12, 16$	31.3
$2 \times 6, 32$	39.3

Table 3. Improved CICE performance for the $320 \times 384 \times 1$ grid on hopper as a function of MPI task count for a 10-day run. Masked halos are active unless otherwise indicated with an asterisk (*). The plus (+) indicates the old Community Earth System Model (CESM) default decomposition and timing.

MPI task count	Blocksize, blocks per task, decomposition	Time (seconds)
16	$20 \times 48, 8$ roundrobin	211.9
16	$40 \times 32, 8$ sectrobin	220.4
16	$20 \times 48, 8$ slenderX1	222.8
16	$20 \times 384, 1$ slenderX1*+	272.5
64	$10 \times 24, 8$ sectrobin	69.6
64	$10 \times 24, 8$ roundrobin	71.1
64	$10 \times 12, 16$ roundrobin	71.8
64	$10 \times 192, 1$ slenderX1*+	91.0
320	$2 \times 96, 2$ slenderX2	21.5
320	$8 \times 6, 8$ sectrobin	25.2
320	$1 \times 48, 8$ slenderX1	25.9
320	$2 \times 192, 1$ slenderX2*+	27.3
1280	$8 \times 6, 2$ spacecurve	10.5
1280	$8 \times 6, 2$ spacecurve*	11.7
1280	$4 \times 6, 4$ spacecurve	11.9
1280	$4 \times 6, 4$ spacecurve*	13.1
1280	$4 \times 6, 4$ sectrobin	14.4
1280	$8 \times 12, 1$ square*+	19.2

Tables 3 and 4 show the current best performance found for a $320 \times 384 \times 1$ displaced pole configuration and a $3600 \times 2400 \times 0.1$ tripole configuration on hopper for several MPI task counts. Tables 3 and 4 also show the default decompositions and baseline timings that were in place before this study. For the $g \times 1$ grid in Table 3, the best performance at lower task counts tends to be decompositions like roundrobin or sectrobin that handle load balance best. At the higher task counts, the cost of halo updates becomes more important and the spacecurve or the cartesian-slender decompositions perform better. The same is true of the $t \times 0.1$ configuration in Table 4. At lower task counts, sectrobin and roundrobin perform well. At higher task counts, spacecurve and sectcart perform best. Compared to the old defaults, CICE performance has been improved between 20% and 45% for the cases

Table 4. Improved CICE performance for the $3600 \times 2400 \times 0.1$ grid on hopper as a function of MPI task count for a 10-day run. Masked halos are active unless otherwise indicated with an asterisk (*). The plus (+) indicates the original default decomposition and timing.

MPI task count	Blocksize, blocks per task, decomposition	Time (seconds)
1200	40×30 , 6 sectrobin	360.9
1200	30×40 , 6 sectrobin	367.2
1200	30×20 , 12 roundrobin	388.8
1200	6×100 , 12 slenderX2* +	582.0
4800	18×25 , 4 sectrobin	107.3
4800	15×15 , 6 sectrobin	110.4
4800	18×25 , 4 spacecurve* +	148.4
18,000	6×6 , 14 spacecurve* +	68.4
18,000	6×10 , 8 sectcart*	70.3
18,000	6×6 , 14 spacecurve	70.9
18,000	8×8 , 8 spacecurve*	73.3
18,000	12×20 , 2, spacecurve*	73.6
18,000	6×5 , 16, spacecurve*	73.9

shown in Tables 3 and 4, except the $t \times 0.1$ case on 18,000 MPI tasks. In general, the masked halo implementation has less impact at relatively high task counts when the spacecurve decomposition is already optimal. For $g \times 1$ (Table 3), the masked halo improved the performance by 10%, while for the $t \times 0.1$ case (Table 4), the masked halo reduced the performance by 4%. This is consistent with the results in Figure 6 where decompositions that were communication intensive benefitted the most from the masked halo algorithm. In addition, the masked halo implementation currently does not mask points along the tripole zipper because the halo algorithm there uses different logic. That shortcoming may differentiate the way the $g \times 1$ and $t \times 0.1$ results behave at high task counts with the masked halo algorithm turned on. New decomposition tools have been implemented in the CICE model scripts in the CESM to automatically predict a reasonable decomposition strategy for any grid and task count, and this tool continues to be refined.

Finally, one important parameter to set correctly is the CICE maxblocks compile-time variable, which specifies the maximum number of blocks on any MPI task. For configurations that do not leverage land block elimination, this value is relatively easy to compute. However, for the spacecurve, roundrobin, and sectrobin decomposition, the maxblocks parameter depends on the number of land blocks that are eliminated. This can be predicted, but in practice, a maxblocks number is usually guessed and then CICE writes, at run time, the number of maxblocks actually needed. If the maxblock value is set too low, the code will abort at initialization. In a few cases, using a max-block value that was too high resulted in a small performance degradation, but this was not universal. In

general, the best recommendation is to set the max-block value to the minimum value needed for any given decomposition.

4. Summary and future work

CICE model performance was evaluated in the CICE model in CESM1.1 on standard global CESM grids. This version was based on CICE v4.1 and the modifications and features described here are available in the CESM1.1 public release and beyond. These modifications are also available in the CICE v5 stand-alone release from LANL in 2013.

The CICE model performance is very sensitive to the parallel decomposition. For a given configuration and processor count, the CICE decomposition can change the model timing by at least a factor of 2 or 3. The block size and maximum blocks per task are compile-time settings that control the decomposition. The decomposition strategy is a run-time setting in CICE. There are currently several different decomposition strategies supported in CICE, including three new decompositions, roundrobin, sectrobin and sectcart, that were added as a result of this study. Each decomposition has a unique set of features that can make it useful in various situations. The main features to consider are balancing CICE work across different tasks, the cost of the halo communication, and the ability to leverage land block elimination upfront. The optimal decomposition varies with grid, hardware, and target task count. At relatively low task counts, load balance dominates the cost and the cartesian-slenderX1, cartesian-slenderX2, roundrobin, and sectrobin decompositions should perform best. At high task counts, the MPI halo communication cost dominates and the cartesian-square and spacecurve decompositions might perform best. If both load balance and communication play relatively equal roles in costs then the cartesian-slenderX1, cartesian-slenderX2, sectrobin, and sectcart decompositions may be better choices. These recommendations are based upon typical CESM global grids. Regional grids may produce different results, as the optimization depends highly on the distribution of sea ice on the grid. The CICE decomposition has minimal impact on the coupler costs in the CESM because the coupler generally runs on a partially overlapping set of tasks and the cost to rearrange the coupling data from the ice decomposition to the coupler decomposition is usually much less expensive than the cost of the CICE model.

A new masked halo feature was added to CICE as a result of this study. This optimization resulted in significant performance improvement. In some cases, the halo communication cost was cut in half and the CICE model timings improved by over 25% overall. The performance benefit depends on the baseline cost and the

decomposition strategy and has the greatest impact on decompositions with relatively expensive halo cost.

In practice, there are literally hundreds of combinations of block sizes and decomposition strategies for any grid and task count. Tools that determine optimum decompositions, generate optimized scaling curves, or identify best task counts are needed. The CESM test framework contains an ice performance test (ICP) that sets up and runs several CICE decompositions for a given configuration and task count. Its output is a summary timing file, but this test can be cumbersome and expensive to run regularly. Argonne National Laboratory researchers are working on a machine-based learning tool to help determine optimal CICE decompositions (Balaprakash et al., 2013). In addition, algorithms in the CESM scripts that automatically set block sizes, maxblocks, and decomposition for a given grid need to be improved, taking into account land block elimination to better identify potential performance sweet spots. Future work might include an off-line performance model of CICE to quickly analyze different decomposition strategies and identify the likely best options for a given machine and grid.

The cost of the CICE model also varies interannually, seasonally, and diurnally as the sea ice distribution and solar angles change. Future work to assess the CICE cost taking into account the temporal sea ice variability could produce new optimizations or decompositions. All tests in this study were run in January conditions. The optimal static decomposition is likely somewhat different in each season. Finally, the CICE model might be a good candidate for dynamic load balance algorithms because sea ice varies across the grid primarily on seasonal timescales, which may be slow enough to offset the overhead associated with any dynamic load balance scheme.

As a result of this detailed study, the changes to the halo implementation, and new decomposition options, the CICE performance has been improved in the CESM by up to 45%. In addition, the results of this study provide a starting point for tuning and optimization of other CICE configurations in the future.

Funding

The CESM project is supported by the NSF and the Office of Science (BER) of the US DOE. NCAR is sponsored by the NSF. This work has been supported by the Office of Science (BER), US DOE, under contracts DE-FC02-97ER62402, DE-FC02-07ER64340, DE-AC02-06CH11357, and DE-AC52-06NA25396. Additional support has been provided by the NSF grant AGS-0856145. This research used resources at the Climate Simulation Laboratory at NCAR's Computation and Information Systems Laboratory (CISL), sponsored by the NSF and resources of the NERSC, which is supported by the Office of Science of the US DOE under contract no. DE-AC02-05CH11231.

References

- Balaprakash P, Alexeev Y, Mickelson S, et al. (2013) Machine-Learning-Based Load Balancing for Community Ice Code Component in CESM. In Proceedings for 11th International Meeting on High-Performance Computing for Computational Science (VECPAR 2014), Eugene, Oregon, July 2014.
- Bitz CM and Lipscomb WH (1999) An energy-conserving thermodynamic sea ice model for climate study. *Journal of Geophysics Research: Oceans* 104: 15669–15677.
- Briegleb BP and Light B (2007) A delta-Eddington multiple scattering parameterization for solar radiation in the sea ice component of the Community Climate System Model. NCAR Tech. Note NCAR/TN- 472 + STR, National Center for Atmospheric Research.
- Dennis JM (2007) Inverse space-filling curve partitioning of a global ocean model. In: *IEEE international parallel and distributed processing symposium*, Long Beach, CA, March 2007, DOI: 10.1109/IPDPS.2007.370215.
- Dukowicz JK and Baumgardner JR (2000) Incremental remapping as a transport/advection algorithm. *Journal of Computational Physics* 160: 318–335.
- Gent PR, Danabasoglu G, Donner LJ, et al. (2011) The Community Climate System Model Version 4. *Journal of Climate* 24(19): 4973–4991.
- Hunke EC and Dukowicz JK (1997) An elastic-viscous-plastic model for sea ice dynamics. *Journal of Physical Oceanography* 27: 1849–1867.
- Hunke EC and Dukowicz JK (2002) The Elastic-Viscous-Plastic Sea Ice Dynamics Model in general orthogonal curvilinear coordinates on a sphere—incorporation of metric terms. *Monthly Weather Review* 130: 1848–1865.
- Hunke EC, Lipscomb WH, Turner AK, et al. (2013) CICE: the Los Alamos Sea Ice Model Documentation and Software User's Manual Version 5.0, LA-CC-06-012.
- Lemieux JF, Knoll D, Tremblay B, et al. (2012) A comparison of the Jacobian-free Newton-Krylov method and the EVP model for solving the sea ice momentum equation with a viscous-plastic formulation: a serial algorithm study. *Journal of Computational Physics* 231: 5926–5944.
- Lipscomb WH and Hunke EC (2004) Modeling sea ice transport using incremental remapping. *Monthly Weather Review* 132: 1341–1354.
- Losch M, Menemenlis D, Campin JM, et al. (2010) On the formulation of sea-ice models. Part 1: Effects of different solver implementations and parameterizations. *Ocean Modelling* 33: 129–144.
- Zhang J and Hibler WD III (1997) On an efficient numerical method for modeling sea ice dynamics. *Journal of Geophysical Research* 102: 8691–8702.

Author biographies

Anthony Craig works as a software engineer for NCAR in Boulder, CO. He has a master's degree in physical oceanography from the University of Washington, but is now more focused on model development, software engineering, and high-performance computing issues in the climate modeling field.

Sheri Mickelson worked as an Application Software Specialist working in both the Mathematics and Computer Science division at Argonne National Laboratory and the Argonne-University of Chicago Computational Institute. She received her bachelor's degree in Computer Science from North Central College. She has worked on CAVE5D, the Fast Ocean Atmosphere Model, and CESM. Her current focus is high-performance computing applications and she is currently working within the Computer and Information System Laboratory and the Earth System Laboratory divisions at NCAR.

As a Scientist at LANL, Elizabeth Hunke develops and maintains the Los Alamos Sea Ice Model, CICE, which is used in numerous climate modeling centers around the world. CICE was developed for global climate studies on highly parallel computing systems, but it is also

used for smaller scale, higher resolution studies of regional climate processes. Although she spends most of her time working in front of a computer in New Mexico's high desert, she has enjoyed sea ice and oceanographic field experience in the Weddell Sea, Antarctica. She serves as a US delegate to the International Arctic Science Committee.

David Bailey has an undergraduate degree in applied mathematics from the University of Waterloo, a master's in Oceanography from the University of British Columbia, and a PhD in Atmospheric and Oceanic Sciences from the University of Colorado. His postdoctoral work was in Oceanography at the University of Washington and the Center for Ocean Land Atmosphere Studies. He has served as the CESM Polar Climate Working Group community liaison at NCAR since 2006.