Scaling the ISAM Land Surface Model Through Parallelization of Inter-Component Data Transfer

Phil Miller*, Michael Robson*, Bassil El-Masri†, Rahul Barman†, Gengbin Zheng‡, Atul Jain†, Laxmikant Kalé*

*Department of Computer Science †Department of Atmospheric Sciences ‡National Center for Supercomputing Applications

University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
{mille121,mprobson,belmasri,barman2,gzheng,jain1,kale}@illinois.edu

***Abstract*—We present the progression of developments necessary to scale the ISAM land surface model from single nodes and small clusters with unusually large per-node memory to much larger systems with more common configurations. These efforts include load balancing, conventional library-based output parallelization to reduce memory load, and parallel-in-time data input. On Hopper, a Cray XE6 machine, the result was strong scaling from 256 cores to 16k cores with an efficiency of 32.9%. On Edison, a Cray XC30 machine, the code strong scales from 256 cores to 16k cores with an efficiency of 51.4%. These large-scale gains, and the associated performance increases at smaller scale, enable greater scientific productivity for the users of ISAM and open the possibilities of increased resolution in time and space and greater physical fidelity for the simulated processes while remaining computationally feasible.**

I. Introduction

Computational climate modeling is a key problem for contemporary high-performance computing. The numerous coupled phenomena that comprise the climate present a uniquely demanding challenge. Various elements have time scales that range from seconds and minutes to years and millennia, and space scales ranging from individual rows of farm fields to the entire globe. Given the limits of interest and expertise, models focusing on just a few effects are typically developed in isolation by researchers specialized in each particular sub-area.

The interactions between different model components are handled either via on-line coupling [1] or consumption of saved model output and processed sensor data from 'upstream' components as input to 'downstream' components. The former can create substantial computational challenges, while the latter can require storage of and access to massive data sets. In this paper, we explore some elements of the off-line case in the Integrated Science Assessment Model (ISAM).

ISAM is a land surface model code used to study biogeophysical and biogeochemical phenomena and their interaction with the larger climate and ecosystem. It couples fluxes of carbon, nitrogen, water, and energy between the near-surface atmosphere, plant growth, and material accumulation above and below ground. ISAM has been used extensively in various modeling and synthesis studies, as described in section II-A, including studies coupled with the Community Earth System Model (CESM). The ISAM code presents two different points for data transfer between components.

The first inter-component data dependence in ISAM arises from the time-dependent climatic variables that serve as boundary conditions for its computations. In an online coupled simulation, such as with other components of CESM [2], these variables become available as they are computed, and may be dependent on results of previous land-surface time steps. In a stand-alone off-line simulation, these are provided in NetCDF files [3], [4] generated and distributed by external groups. Their treatment is discussed in section V.

The second point of inter-component data transfer occurs between separate runs performing different stages of model spin-up. Section II describes the distinct biogeophysics and biogeochemistry model components that make up ISAM. During model 'spin-up', these components are run independently for different lengths of simulated time before coupling them to conserve computational resources. In order to set initial conditions for each of these runs, large portions of model state from the preceding run must be preserved. The optimization of that storage operation is discussed in section IV.

This paper also describes the other changes necessary to enable ISAM to scale to large supercomputer systems. Section III discusses the load balancing issues in distributing model grid points among processors in the parallel system. Section VI discusses remaining impediments to large-scale execution. The end result of this effort is that the model is able to strong-scale a whole-Earth land surface simulation from 256 to 16k cores of NERSC's Edison Cray XC30 system with over 50% parallel efficiency, and with a clear path to further improvement.

II. Background

A. ISAM: Integrated Science Assessment Model

The Integrated Science Assessment Model (ISAM) is a land surface model which is a fully coupled carbon-nitrogen model that simulates carbon, nitrogen, water, and energy fluxes at $0.5^\circ \times 0.5^\circ$ spatial resolution at multiple temporal resolutions, ranging from half hourly to yearly [5]. Each grid is occupied by combination of plant functional types (PFTs), bare ground, and glaciers for a total of 28 different PFTs types [6]. The model is driven by half hourly or hourly climatic variables:

mean surface air temperature, precipitation, pressure, incoming shortwave and long-wave radiation, wind speed, and specific humidity. In ISAM, there are 10 hydrological and thermal active layers, 5 hydrologically inactive and thermally active bedrock layers, 5 snow layers, 7 vegetation carbon pools and 8 litter and soil organic matters pools.

ISAM has two main components: (1) a biogeophysical module representing sunlit-shaded photosynthesis schemes [7], energy, and soil/snow hydrology [8], [9], [7] and (2) a biogeochemical module, where assimilated carbon is allocated to vegetation, litter, and soil carbon pools [5], [7], [10]. The carbon cycle and nitrogen cycle are coupled together. The nitrogen cycle accounts for the dominant physical processes, including fixation, deposition, mineralization, leaching, immobilization, and nitrification and denitrification [11], [10].

ISAM is applied to examine the impacts of changing CO_2 in the atmosphere, fire, and land use change on terrestrial ecosystems functions [12], [13], [14]. ISAM has also been used to describe the dynamic of the terrestrial biosphere carbon and nitrogen [11], [10] and was a part of all five IPCC Working Group I assessment reports on science of climate change. ISAM has also participated in the Modeling and Synthesis Thematic Data Center (MAST-DC) study as part of the North American Carbon Program (NACP) and the Multi-scale Synthesis and Terrestrial Model Intercomparison Project (MSTMIP) to quantify and understand spatial and temporal distributions of carbon sources, sinks, and inventories by synthesizing NACP data and models, from sites to regional/continental scales [15], [16], [17], [18]. Also, ISAM was used to study the impact of elevated CO_2 on ecosystem carbon and nitrogen cycles part of the Free-Air CO_2 Enrichment (FACE) experiment [19].

In order to better understand how the interactions among the climate, the land-surface, and human activity can amplify or mitigate the pace of climate change, we coupled the current version of ISAM with the Community Earth System Model (CESM1), to develop an Earth System Modeling framework: CESM-ISAM. The CESM-ISAM retains the existing land surface model in CESM, i.e., the Community Land Surface Model 4 (CLM4), and allows both the ISAM and CLM to choose from all of the existing configurations available in CESM. Additionally, the resulting framework has been designed to incorporate multiple land surface models into the CESM, by adopting a flexible approach to coupling through the flux coupler. The purpose of this general modeling framework is to carry out equivalent climate simulations using multiple land surface models with the rest of the component models being the same, allowing a direct comparison of the effects of different land surface representations on corresponding feedbacks to climate change. Such a modeling framework establishes multiple opportunities to investigate the role of varying representation of land surface processes (such as from biogeophysics and biogeochemistry) on coupled land-atmosphere interactions.

B. Model Spin-Up

ISAM requires the use of long spin up times of over 20,000 simulated years to reach the desired steady state conditions, especially in the soil biogeochemical cycles. For such long model integration periods, the spinning up of fully coupled biogeophysics-biogeochemistry often requires extensive (and prohibitive) computational resources, and provides a major challenge in the modeling of cold region biogeochemistry using current land surface models. Thus, ISAM uses an innovative method of model spin up, where (1) first only the model biogeophysics is spun up using prescribed nitrogen-limitation, typically for approximately 150 years to achieve steady state in canopy carbon fluxes, (2) subsequently only the soil biogeochemistry is spun up for 20,000 model years, using the steady state conditions achieved in the first stage, and (3) finally, the coupled biogeophysics-biogeochemistry are spun-up for 150 years, using steady state conditions from the first and second steps. ISAM's capability to allow sequential spin-up of physical and biogeochemical cycles tremendously reduces the computational expense, allowing for equivalent model integration of greater than 10,000 years. Such a decoupled method of model spin up is effective, because the spin up time scales of soil biogeophysics are much shorter than that of biogeochemistry: typically in the order of 100s of years, and hence only the latter need to be spun up for long timescales.

C. Computational Structure

ISAM operates over a grid of points overlaid on the portions of the Earth's surface that lie over land. The computation at each point has no interaction with the computation at any other point. Thus, its execution should ideally benefit from the 'pleasantly parallel' nature of the problem [20]. The lack of inherent communication and the freedom to assign the work of any point to any processor eases parallelization of the core elements of the model, and indeed the work described in sections III and V exploit these traits heavily.

As the rest of this paper shows, after optimization, the computational work of the model at each point dominates the runtime up to a few thousand processors. However, at larger scales, other necessary elements of the code exert a stronger influence. Serial bottlenecks, in the sense described by Amdahl's law, become noticeable. Global operations that are not so trivially parallelized encounter scaling limitations that overwhelm the achieved speedup in the main computation.

D. Coupling to Atmospheric Conditions

At each of ISAM's timesteps, the code uses the corresponding atmospheric state just above the surface as a boundary condition at each simulated grid point. We focus on the offline case, where these conditions are provided from data sets generated from observations and separate atmospheric models. Currently, the model supports NCEP [21], CRU-NCEP [22], and NLDAS [23] climate data sources. These data sources provides climate variable information for temperature, specific humidity, wind speed, precipitation rate, surface pressure, incident radiation, and CO_2 concentration.

Because the atmospheric data does not necessarily match the land surface model in spatial resolution or alignment, it must be interpolated from the points at which it is provided to the points at which the simulation runs. This interpolation is done online to allow ISAM to be configured at runtime, without requiring a long pre-processing step. The interpolation represents a small fraction of the overall computation.

E. Synchronization Cost of Imperfect Load Balance

The various points at which a processor must wait for other processors to coordinate a synchronous global operation present opportunities for load imbalance to negatively impact performance. At these points, the waiting time is determined by the execution time of the most heavily loaded processor. The consequent loss of performance can be mitigated by a combinations of improving load balance and removing the need for synchronization. We can measure the overall cost to performance using the *imbalance time* metric [24]. All 'Idle' time displayed in our figures is attributable to imbalance time, since there is no operation-dependent communication latency or other source of underutilization.

There are several distinct sources of potential load imbalance in ISAM. The different sets of points assigned to different processors may generate different cumulative loads over various time scales (§ III). Some operations may be performed serially on a single processor while others wait (§ V-B,VI-A). Contention for shared resources may induce imbalance where it is not otherwise inherent in the operation being performed (§ VI-C).

There are also synchronous operations that can be reduced. They can be made less frequent directly (§ V-A) and by having the code do more with each one (§ V-C).

F. Experimental Setup

Our scaling experiments reported in sections IV–VI were run on the Cray supercomputers hosted at NERSC, Hopper and Edison. Hopper is a Cray XE6 with 6,384 nodes each containing a pair of 12-core AMD 'Magny Cours' processors running at 2.1 GHz. Edison is a Cray XC30 with 5,576 nodes each containing a pair of 12-core Intel 'Ivy Bridge' processors running at 2.4 GHz. Edison's cores support 2-way HyperThreading SMT, but our experiments were run with a single thread per core. Both systems are served by Lustre parallel filesystems, on which all files relevant to our experiments were stored. On both machines, we used the Intel Fortran Compiler. Binaries for runs on Hopper were compiled with version 13.1 and binaries for runs on Edison were compiled with version 14.0.

All of our figures show the best results on each scale on each machine. Due to run-to-run variability (from node placement and contention with other jobs for network and filesystem resources [25]), some experimental runs were up to 16% slower than the speeds displayed.

III. Computational Load Balance

The phenomena modeled by ISAM present many opportunities for load imbalance between individual grid points. The load of each point can vary on predictable temporal cycles with diurnal and seasonal effects, such as solar irradiance and growth periods. They can also vary somewhat predictably by geography. Persistent deserts present minimal plant activity and thus require little work to compute their effect. High latitude points closer to the poles must account for the development and step-by-step evolution of snow layers that are less prevalent in more tropical regions. These temporal and geographical variations are mediated by less-predictable climatic effects that determine the precise degree to which the relevant phenomena occur at fine time scales.

In initial versions of ISAM, land surface points were mapped to MPI ranks in a uniform blocked fashion: the total number of points being simulated was divided by the number of ranks in the job, and each rank took responsibility for a consecutive chunk of points. As generated from the data set representing which points on Earth's surface were land (rather than open water), the points were ordered along successive lines of latitude. Each core was thus likely to receive points that were geographically nearby and thus both similar in systematic spatial work characteristics and closely correlated in temporal cycles and climate variations.

The result of this structure was that entire regions of the planet presenting high workload would be assigned to one set of cores, while regions presenting much less work would be assigned to others. The consequent aggregate utilization of each processor was highly varied. This can be seen in figure 1a, generated from traces of a 24-core run in the Projections tool [26]. Since the code incurs regular global synchronization, even short-term dynamic load variation causes performance to suffer (§ II-E). Otherwise, each run's time to solution would be determined simply by the maximum cumulative load on any core.

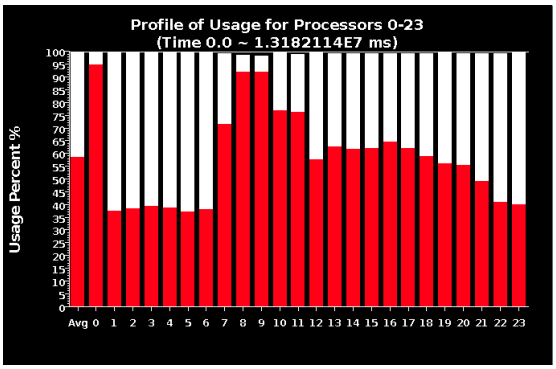
A. Round Robin

To address this issue, the code was adapted to distribute the points cyclically, in a round-robin fashion. The new mapping is illustrated in figure 2. This ensures that the points of any given region are spread across many cores, and that each core hosts points from distinct regions. This results in mixing high-load points with low-load points, such that they tend to average out. The core utilization after switching to this mapping can be seen in figure 1b.

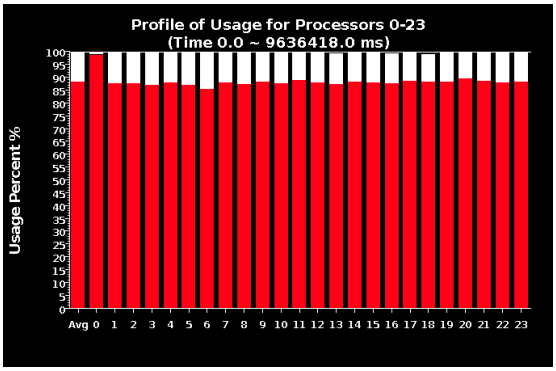
Applying this optimization enabled strong scaling from a single node to a few dozen nodes as in a small cluster. Benchmark results at this stage are presented in figure 3. These measurements were taken on the Jaguar Cray XT5 system at Oak Ridge National Laboratory. The model was run for 1 simulated year.

IV. Parallel Output to Reduce Memory Footprint

As described in section II-B, the biogeochemistry spinup is parameterized by preliminary steady-state values of the biogeophysics portion of the model. This requires saving the values of the coupling variables at every simulated point over a sufficient span of time steps to provide accurate results. A



(a) Before: block mapping



(b) After: round-robin mapping

Fig. 1. Processor utilization of each rank visualized in the Projections tool

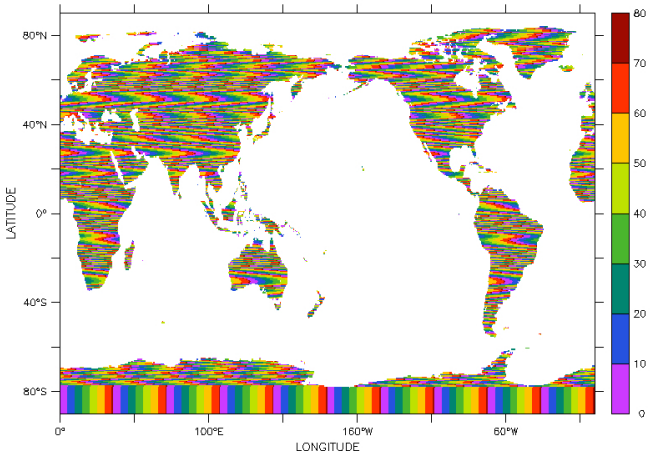


Fig. 2. Round-robin mapping of land surface grid points across 72 cores. Process rank is represented by color of each point.

Cores	Time (minutes)
128	76
256	46
512	34
1024	35
2048	33

Fig. 3. Preliminary parallel performance: execution time of ISAM for 1 simulated year on Jaguar XT5 with round-robin mapping of points to cores

similar data set is also necessary to checkpoint the biogeophysics component's state, so that it can be restarted after system failures or job length expiration.

The largest of these elements are a set of 10 variables that are recorded for each of the 28 PFTs at every point with a weekly resolution over the course of a simulated year. At a resolution of 360×720 (i.e. $0.5°$) and recorded in double precision, each variable occupies 2.8 gigabytes.

The initial implementation and usage of ISAM was on a cluster of large-memory nodes, which could gather the entire data for a given variable in a single processor's memory and write it serially. In tested implementations, the gather and write operations could each transiently consume up to double the final buffer size, requiring over 5 gigabytes of free memory available to a single process beyond the normal per-processor model state. On systems built from nodes with less memory, this process would exhaust available memory entirely and cause job failure.

This demand for large-memory nodes is problematic for two reasons. The first is that most larger clusters and supercomputers are built from nodes with relatively modest amounts of memory per node, since a large proportion of applications make lighter demands and the cost of additional memory in each node is preferentially spent to obtain additional nodes. The second is that even where large memory nodes are available, they are a more heavily subscribed resource, prospectively delaying job start and thus extending the time-to-completion of each job. For instance, Hopper offers 6,000 nodes with 32 GiB of RAM, but only 384 nodes with 64 GiB.

By parallelizing the output of these variables across multiple nodes, we reduce the maximum per-node memory footprint of the output operation. A reduced footprint thus enables execution on more plentiful computational resources and more rapid scheduling of jobs on those resources.

To implement parallel output of these variables, we adapted the code to use the Parallel netCDF library [27]. We maintain the same netCDF file format as the serial implementation to allow easy verification of the new code and compatibility with other existing tools that are used to operate on the resulting intermediate files. However, for the sake of providing the library with contiguous blocks of output from each process instead of single strided points, we transposed the dimension order of the arrays as written.

On Hopper, using version 1.2.0 of Parallel netCDF, we were able to write the 25 GB making up a complete set of these largest variables to the Lustre 'scratch' filesystem in 17 seconds, for an overall bandwidth of 1.46 GB/s. While this is only a small proportion of the filesystem's peak 35 GB/s bandwidth, it is sufficiently fast to avoid this phase of execution becoming a substantial bottleneck.

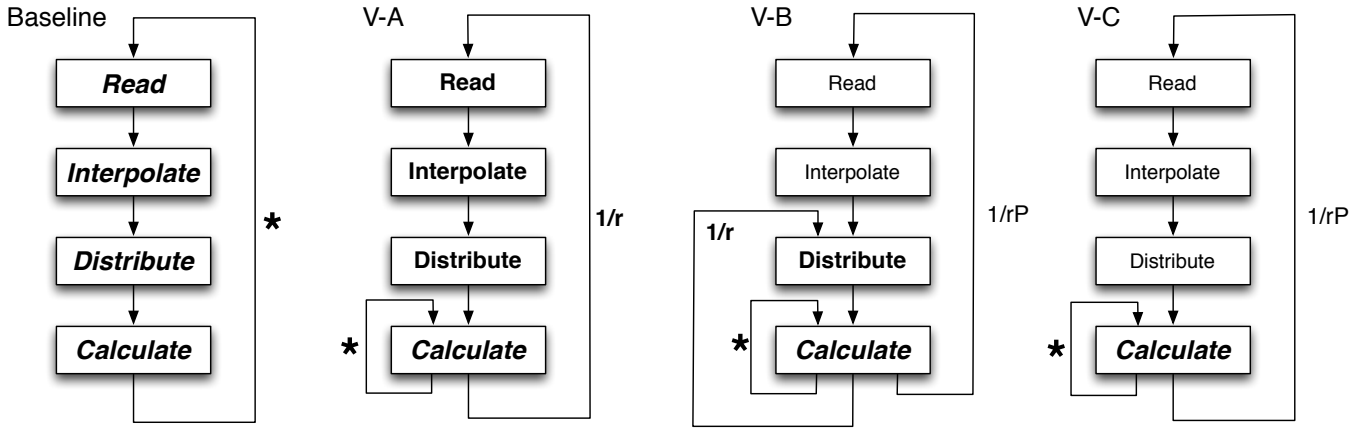


Fig. 4. An illustration of the process by which various versions of ISAM acquire and consume atmospheric input data. Each flow chart represents one version of the code, with various optimizations implemented. The optimizations are described in the section corresponding to the label on each flow chart. In the baseline version, input is read, interpolated, and distributed in each step (shown by '*' loops). Later versions reduce redundant work by a factor of $3 \leq r \leq 12$, the ratio of model time steps to atmospheric input time resolution ($1/r$ loops). Finally, non-computational work is parallelized in two stages ($1/rP$ loops).

V. Input Optimization

The basic structure of the climate forcing data input process involves three steps. A process reads the appropriate point in the time series provided by the input files, using the NetCDF library [3], [4]. It then computes the spatial interpolation to the simulated land surface points. Finally, the interpolated data are distributed to the processes according to which grid points they are responsible for, as described in section III. This structure, and the changes to it described in this section, are illustrated in figure 4.

The initial design of this input presented many impediments to scalability. In the remainder of this section, we discuss how these limitations have been largely eliminated. All data are shown in figures 5 and 6. The individual curves and bars are keyed by the following subsection headings whose optimizations they depict, and 'R' is used for the round-robin mapping described in section III.

A. Matching Atmospheric Timestep with Model Timestep

The ISAM model is typically run with a timestep of 30–60 minutes. The atmospheric data are provided at time intervals of 3–6 hours. In the initial implementation of ISAM, the input process read the most recent atmospheric data from the source files, interpolated it, and distributed it, for every model timestep. Thus, the latency of filesystem access, interpolation, and `MPI_Scatterv` was on the critical path of successive model timesteps. Given that the same data would be provided for 3–12 steps in a row, this repetition was both redundant and created excess synchronization.

By modifying ISAM to reuse already-prepared data, we improved performance by a factor of $1.7\times$ on 1024 ranks of Hopper and $1.2\times$ on 1024 ranks of Edison. This improvement comes from both reduced time spent performing collectives, and reduced imbalance time waiting on heavily-loaded cores to reach each collective.

On 1024 cores of Hopper, where the round-robin and present optimized versions of the code obtain their best performance, the optimized code spends 88% less CPU time performing collectives and 24% less CPU time idling. The decline in collective time accounts for 62% of the $1.7\times$ speedup and the decline in idle time accounts for a further 34% of the speedup.

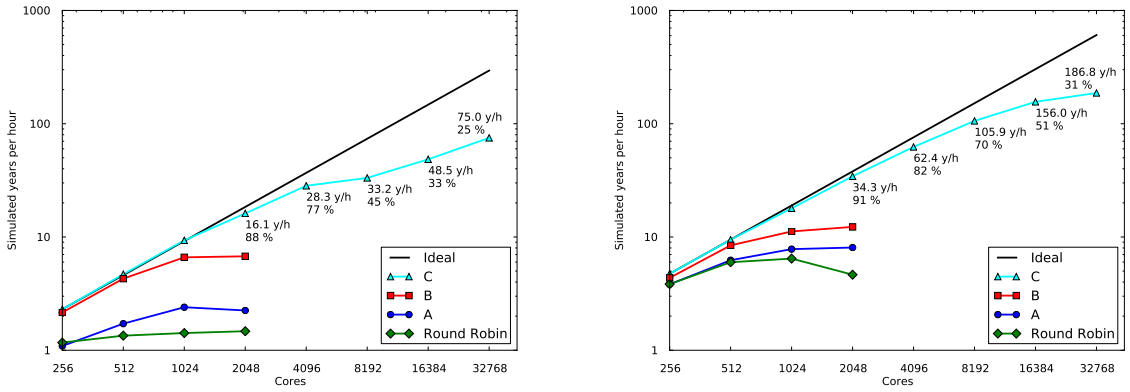
On 1024 cores of Edison, where the round-robin and present optimized versions of the code also obtain their best performance, the optimized code spends 30% less CPU time performing collectives and 18% less CPU time idling. The decline in collective time accounts for 64% of the $1.2\times$ speedup and the decline in idle time accounts for a further 33% of the speedup.

B. Parallel-In-Time Reading and Interpolation

With input data read from the filesystem every few steps, the time per step scales poorly due to an Amdahl's law bottleneck on the time to access the filesystem and interpolate the data. Additionally, contemporary supercomputers offer high-bandwidth parallel filesystems to support their computational capabilities. By reading input data using only a single rank, ISAM was limited to the bandwidth of a single node.

Thus, our next optimization to ISAM's input process is to read and interpolate many steps worth of input data in parallel. At model timesteps where data must be read, each process reads and interpolates data for a step computed by incrementing the current timestep by its rank. At each subsequent step, the responsibility for distributing data cycles across the ranks until every rank has served as the root once.

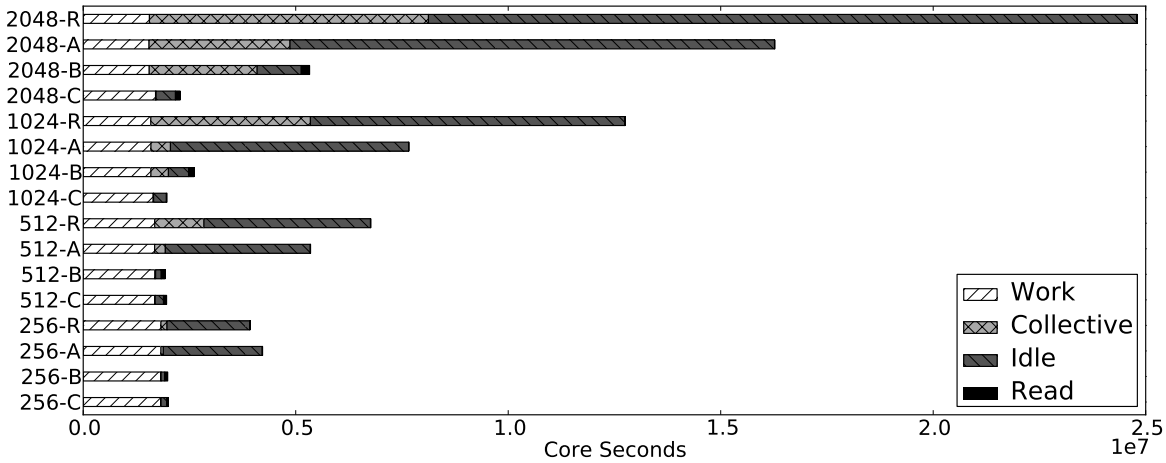
In theory, this can reduce the elapsed wall time spent on reading and interpolation by $\mathcal{O}(P)$, since P such steps are performed in parallel. This is potentially limited by available bandwidth both in accessing the file data from the filesystem and in interpolating it in memory. At larger scales, we observe



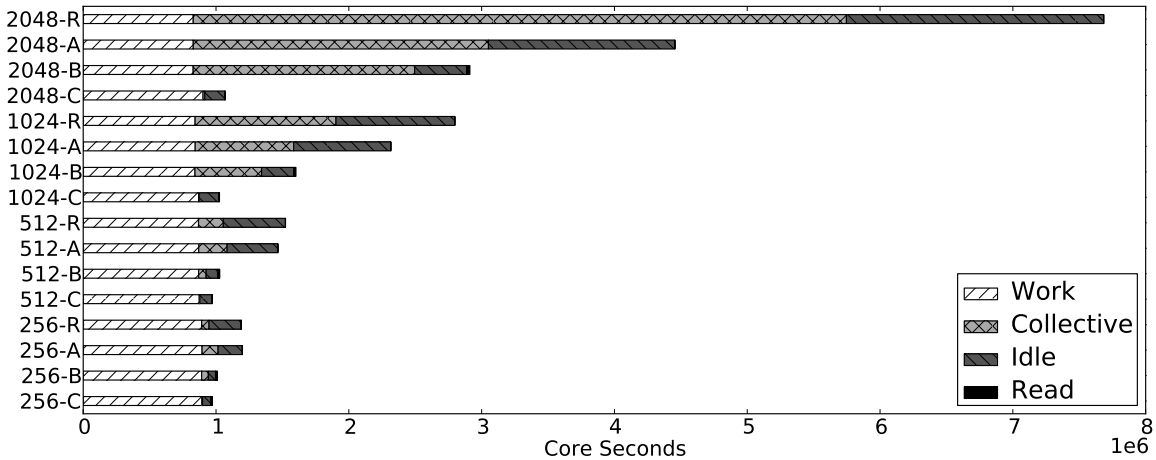
(a) Hopper XE6

(b) Edison XC30

Fig. 5. Overall scaling of ISAM as successive optimizations are applied to the input process for the climate forcing data. The graphs show years of simulated time per hour of execution wall time. Higher is better. Runs were for 5 years of simulated time. The legend for each curve refers to the subsections of section V describing the corresponding optimizations. Labeled points show precise performance values and parallel efficiency relative to the most optimized code version 'C' on 256 cores.



(a) Hopper XE6



(b) Edison XC30

Fig. 6. Breakdown of core-seconds spent by ISAM on different activities, as a function of scale and applied optimizations of the climate forcing data input process. Runs were for 5 years of simulated time. Lower is better.

this effect, as described in section VI-C.

The improvement provided by this optimization over that described in section V-A is $2.76\times$ on 1024 cores of Hopper and $1.3\times$ on 1024 cores of Edison. In both cases, the reduction in idle time accounts for the bulk of the improvement. On both systems, this optimization allows the code to continue to gain performance at scales up to 2k cores, with efficiencies of 39% and 34% respectively, relative to the 256 core baseline.

Note that the memory load imposed by this adaptation scales with the number of ranks per node, rather than the number of ranks in the entire job. The code does not distinguish between rank 0 and all other ranks; thus they all have the capacity to read and interpolate input. In a setting where the total memory on a node is insufficient to buffer a step's input per rank, we could adjust the scheme to only read and interpolate on every kth rank instead. This simply adjusts the above theoretical impacts by a constant factor of k, without changing the conclusion of improved scalability.

C. Simultaneous Distribution of Multiple Steps

Having minimized idle time by fully parallelizing the reading and interpolation steps, the largest non-work portion of the execution time at the scaling limit of the code from section V-B is spent in collectives. On 2048 cores, these consume 46% of CPU seconds on Hopper and 57% of CPU seconds on Edison. On both systems, the increases in collective times account for the bulk of increased time relative to runs on 1024 cores.

To overcome this impediment, we observe that at the first scatter operation after climate forcing data is read and interpolated, the P processors each have data available for an upcoming timestep. However, in each scatter, only the cyclically selected root processor actually provides it. This misses a substantial opportunity for increased parallelism in usage of network resources.

We take advantage of this opportunity by converting the per-step `MPI_Scatterv` operation to an `MPI_Alltoallv` operation performed every P time steps. Rather than spatially scattering data representing the climate forcing at a single point in time, we now transpose the data from its provided temporal distribution (each core sends a distinct timestep for every point) to a spatial distribution (each core receives the time series for the points it owns). Once this is done, each core can independently execute P time steps with no communication.

At first glance, this pre-distribution of input data may seem to dramatically increase memory usage on every node. However, this is not the case. To see why, we first observe that the additional memory consumption is a constant, independent of P. Suppose there are n points in total, and each one requires b bytes of memory for a single time step's climate data. Each core is responsible for n/P of those points. The data each core reads from disk as in section V-B is bn. In the transposition, each core receives the bn/P bytes for one future time step from each of the P cores. Thus, the total received data is just bn – exactly as much as every core read from disk. For the NCEPQ climate data set, $b = 24$ and $n = 192 \times 94 = 18,048$, totaling 423 kilobytes. For the CRU_NCEP data set, $b = 32$ and $n = 720 \times 360 = 259,200$, totaling 8 megabytes.

The effects of this optimization are striking. Where previously roughly half the execution time was spent in collectives at just 2k cores, this optimization reduces that time to less than 1% on both systems. Additionally, idle times also fell by over 50% on both systems, due to the longer period between synchronization points and greater opportunity for dynamic load variation to average out. Moreover, read times (though representing only a small proportion of execution) also fell substantially because of this optimization. We conjecture that this decrease is due to reduced contention when accessing the filesystem, since different cores can reach this phase across a wider timespan, as opposed to nearly simultaneously. Overall, this provides a $2.4\times$ speedup on 2k cores of Hopper, and a $2.9\times$ speedup on 2k cores of Edison. It also allows us to scale with continued speedups to 32k cores.

D. Summary

From our baseline code, we have obtained speedups of $6.58\times$ on 1024 cores of Hopper and $2.78\times$ on Edison. With all of the optimizations applied, we strong scale from 256 cores to 2048 process with an efficiency of 88% on Hopper and 91% on Edison.

VI. Scaling Discussion

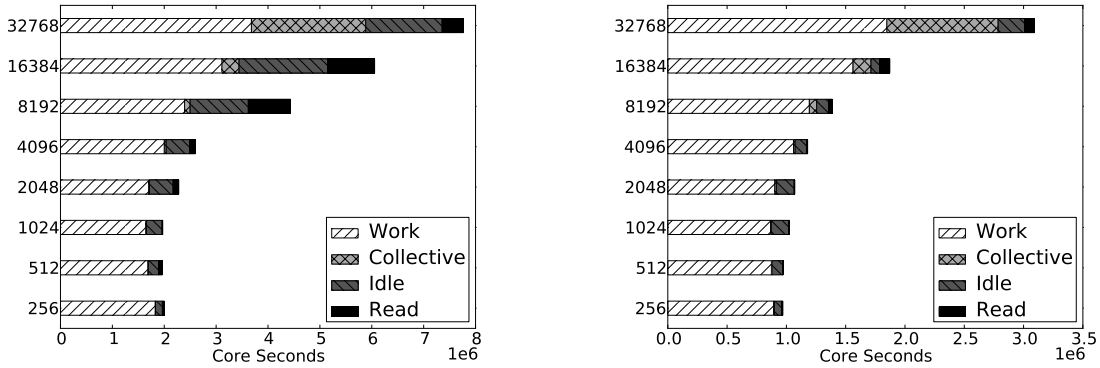
As we reach scales larger than 2k cores with the code from section V-C, we can see substantial increases in the work, collective, and idle times, and smaller increases in read time as shown in figure 7. In this section, we characterize why these remaining impediments occur and how the code may be changed to address them.

A. Per-Year Serial Work

At larger scales, the observed increase in work and idle time is apparently proportional to the number of cores in the job. Finer-grained timing instrumentation in the code indicates that this additional time must occur at the end of one simulated year and the beginning of the next, since mid-year months and days scale near-perfectly. A linear fit to the work time as a function of core count (including runs spanning different numbers of simulated years, not shown) indicates serial or replicated work of approximately 6 seconds of wall time per simulated year, with $R^2 = 0.98$. Several portions of the code test for whether they are operating on the first or last day or timestep of a year, and adjust their behavior accordingly. To scale further, and particularly to obtain speedups approaching or beyond the theoretical limit of $3600/6 = 600$ simulated years per hour, these pieces of the code will have to be improved.

B. Collectives

At 16k cores the code spends just 5% of its time in collectives on Hopper and 8% on Edison. At 32k cores, we see the time spent in collectives increase dramatically. These increases account for almost the entire difference in utilization from 16k to 32k cores. There are several potential causes



(a) Hopper XE6

(b) Edison XC30

Fig. 7. Breakdown of core-seconds spent by ISAM on different activities, as a function of scale for the most optimized version of the climate forcing data input process. Runs were for 5 years of simulated time. Lower is better.

Cores	Points Read	Max Time (s)	Bandwidth (GiB/s)
32,768	14,600	9.89	11.40
16,384	14,600	9.49	11.89
8,192	8,192	5.92	10.70
4,096	4,096	3.68	8.60
2,048	2,048	2.74	5.78
1,024	1,024	2.51	3.15
512	512	2.47	1.60
256	256	1.53	1.29

Fig. 8. Effective read bandwidth at various scales for the five year sample run. The 'Project' GPFS filesystem from which this data was read has a peak bandwidth of 40 GB/s [30].

of this slowdown. In general, the all-to-all algorithms in use may incur contention for message injection at the network interfaces and for links within the network. In particular, the slowdown on Edison may be explained by forced usage of slower 'rank-3' links between groups of nodes in the Cray XC30 'Dragonfly' network topology [28]. This may be mitigated by improved general all-to-all algorithms for such topologies [29] or by implementation of a more specialized transpose operation suited to the needs of ISAM.

C. File System Contention

In order to analyze the effects of a small number of cores with long read times on the scalability of the application we performed a series of small experiments on Edison by modifying our original timing infrastructure to measure each MPI process's individual read time. The results are summarized in figure 8, and presented in histograms in figure 9.

As we scale the code, we see a greater spread in the read time length, whose maximum increases sub-linearly. The improved bandwidth with larger scale is evidence that the reads can effectively use the parallel filesystem. We can see that a large fraction of the cores finish rapidly at larger scales. However, they are forced to wait on the slower cores before finishing the collective that distributes the data. This increases our observed idle time.

Some of this variability can be attributed to the increased sample size (since we are increasing the numbers of cores measured as we scale). However, as we scale our application, more cores are trying to access the shared file system simultaneously. This in turn greatly increases contention which is another contributing factor in the measured idle time increase.

Thus, it may be desirable to limit the number of cores reading simultaneously in order to mitigate contention. As a modification of the technique described in section V-B, the simple use of a subset of Q cores for parallel reading and interpolation would reduce the CPU time spent waiting for read data by $\mathcal{O}(Q)$ relative to the code in section V-A. The associated collectives would occur more frequently by a factor of P/Q, but would each convey proportionally less data. The more frequent synchronization that they impose may present a larger cost (§ II-E). Thus, with this technique an appropriate tradeoff would have to be made for the machine, job size, and system conditions.

A more involved approach would overlap different subsets of cores reading and conveying data simultaneously. This design could take many possible shapes, but would likely benefit from the asynchronous `MPI_Ialltoallv` collective specified in the MPI-3 standard [31].

VII. Conclusion

We have presented the progression of developments necessary to scale the ISAM land surface model from single nodes and small clusters with unusually large per-node memory to much larger systems with more common configurations. These efforts included load balancing, conventional library-based output parallelization to reduce memory load, and parallel-in-time data input. On Hopper, the result was strong scaling from 256 cores to 16k cores for a speedup of $21\times$, giving an efficiency of 32.9%. On Edison, the code exhibits a strong-scaling speedup from 256 cores to 16k cores of $32.9\times$, for an efficiency of 51.4%. These large-scale gains, and the associated performance increases at smaller scale, will enable greater scientific productivity for the users of ISAM and open the possibilities of increased resolution in time and space

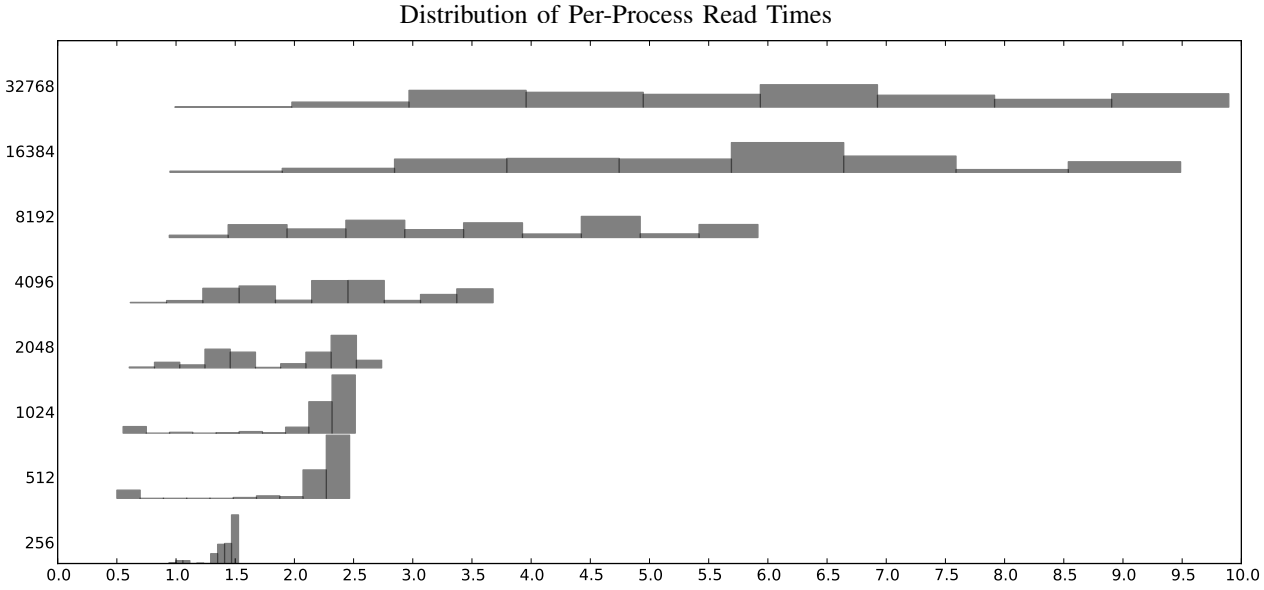


Fig. 9. The distribution of individual rank read times as a function of scale on Edison for the most optimized version of the climate forcing data input process. At each scale, we performed three runs within the same job partition, and selected the run with the highest maximum time. From the selected run, the histogram bin size was calculated as $(\max - \min)/10$. For our five year sample run there are only 14,600 steps to be read and distributed. Thus, we have adjusted percentages of the remaining bins accordingly (i.e. out of 14,600).

and greater physical fidelity for the simulated processes while remaining computationally feasible.

VIII. Acknowledgements

The authors would like to thank Nikhil Jain for advice, support, and some helpful literature references. This work was supported by funding under U.S. Department of Energy grant DE-SC0006706. This research used resources of the National Energy Research Scientific Computing Center (NERSC), which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. This research used resources of the Oak Ridge Leadership Computing Facility at the 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.

References

[1] A. P. Craig, M. Vertenstein, and R. Jacob, "A new flexible coupler for earth system modeling developed for CCSM4 and CESM1," *International Journal of High Performance Computing Applications*, vol. 26, no. 1, pp. 31–42, February 2012.

[2] R. Barman, F. M. Hoffman, D. M. Lawrence, Y. Song, P. Meiyappan, A. K. Jain, R. L. Jacob, and M. Vertenstein, "Studying uncertainties in climate-terrestrial biogeochemical feedbacks in the northern high latitudes using a flexible earth system modeling framework," in *AGU Fall Meeting Abstracts*, vol. 1, 2011, p. 04.

[3] R. Rew and G. Davis, "NetCDF: an interface for scientific data access," *Computer Graphics and Applications, IEEE*, vol. 10, no. 4, pp. 76–82, 1990.

[4] S. A. Brown, M. Folk, G. Goucher, R. Rew, P. F. Dubois *et al.*, "Software for portable scientific data management," *Computers in Physics*, vol. 7, no. 3, pp. 304–308, 1993.

[5] B. El-Masri, R. Barman, P. Meiyappan, Y. Song, M. Liang, and A. K. Jain, "Carbon dynamics in the Amazonian Basin: Integration of eddy covariance and ecophysiological data with a land surface model," *Agricultural and Forest Meteorology*, vol. 182183, no. 0, pp. 156–167, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0168192313000658

[6] P. Meiyappan and A. Jain, "Three distinct global estimates of historical land-cover change and land-use conversions for over 200 years," *Frontiers of Earth Science*, vol. 6, no. 2, pp. 122–139, 2012. [Online]. Available: http://dx.doi.org/10.1007/s11707-012-0314-2

[7] Y. Song, A. K. Jain, and G. F. McIsaac, "Implementation of dynamic crop growth processes into a land surface model: evaluation of energy, water and carbon fluxes under corn and soybean rotation," *Biogeosciences Discussions*, vol. 10, no. 6, pp. 9897–9945, 2013. [Online]. Available: http://www.biogeosciences-discuss.net/10/9897/2013/

[8] R. Barman, A. K. Jain, and M. Liang, "Climate-driven uncertainties in modeling terrestrial gross primary production: a site-level to global scale analysis," *Global Change Biology*, 2013. [Online]. Available: http://dx.doi.org/10.1111/gcb.12474

[9] ——, "Climate-driven uncertainties in modeling terrestrial energy and water fluxes: a site-level to global scale analysis," *Global Change Biology*, 2013. [Online]. Available: http://dx.doi.org/10.1111/gcb.12473

[10] X. Yang, V. Wittig, A. K. Jain, and W. Post, "Integration of nitrogen cycle dynamics into the Integrated Science Assessment Model for the study of terrestrial ecosystem responses to global change," *Global Biogeochemical Cycles*, vol. 23, no. 4, 2009. [Online]. Available: http://dx.doi.org/10.1029/2009GB003474

[11] A. Jain, X. Yang, H. Kheshgi, A. D. McGuire, W. Post, and D. Kicklighter, "Nitrogen attenuation of terrestrial carbon cycle response to global environmental factors," *Global Biogeochemical Cycles*, vol. 23, no. 4, 2009. [Online]. Available: http://dx.doi.org/10.1029/2009GB003519

[12] A. K. Jain, H. S. Kheshgi, and D. J. Wuebbles, "A globally aggregated reconstruction of cycles of carbon and its isotopes," *Tellus B*, vol. 48, no. 4, pp. 583–600, 1996. [Online]. Available: http://dx.doi.org/10.1034/j.1600-0889.1996.t01-1-00012.x

[13] A. K. Jain and X. Yang, "Modeling the effects of two different land cover change data sets on the carbon stocks of plants and soils in concert with CO_2 and climate change," *Global Biogeochemical Cycles*, vol. 19, no. 2, 2005. [Online]. Available: http://dx.doi.org/10.1029/2004GB002349

[14] A. K. Jain, Z. Tao, X. Yang, and C. Gillespie, "Estimates of global biomass burning emissions for reactive greenhouse gases (CO, NMHCs, and NOx) and CO_2," *Journal of Geophysical Research: Atmospheres*, vol. 111, no. D6, 2006. [Online]. Available: http://dx.doi.org/10.1029/2005JD006237

[15] D. Huntzinger, W. M. Post, Y. Wei, A. Michalak, T. O. West, A. Jacobson, I. Baker, J. M. Chen, K. Davis, D. Hayes *et al.*, "North American Carbon Program (NACP) regional interim synthesis: Terrestrial biospheric model intercomparison," *Ecological Modelling*, vol. 232, pp. 144–157, 2012.

[16] T. Keenan, I. Baker, A. Barr, P. Ciais, K. Davis, M. Dietze, D. Dragoni, C. M. Gough, R. Grant, D. Hollinger, K. Hufkens, B. Poulter, H. McCaughey, B. Raczka, Y. Ryu, K. Schaefer, H. Tian, H. Verbeeck, M. Zhao, and A. D. Richardson, "Terrestrial biosphere model performance for inter-annual variability of land-atmosphere CO_2 exchange," *Global Change Biology*, vol. 18, no. 6, pp. 1971–1987, 2012. [Online]. Available: http://dx.doi.org/10.1111/j.1365-2486.2012.02678.x

[17] A. D. Richardson, R. S. Anderson, M. A. Arain, A. G. Barr, G. Bohrer, G. Chen, J. M. Chen, P. Ciais, K. J. Davis, A. R. Desai, M. C. Dietze, D. Dragoni, S. R. Garrity, C. M. Gough, R. Grant, D. Y. Hollinger, H. A. Margolis, H. McCaughey, M. Migliavacca, R. K. Monson, J. W. Munger, B. Poulter, B. M. Raczka, D. M. Ricciuto, A. K. Sahoo, K. Schaefer, H. Tian, R. Vargas, H. Verbeeck, J. Xiao, and Y. Xue, "Terrestrial biosphere models need better representation of vegetation phenology: results from the North American Carbon Program site synthesis," *Global Change Biology*, vol. 18, no. 2, pp. 566–584, 2012. [Online]. Available: http://dx.doi.org/10.1111/j.1365-2486.2011.02562.x

[18] K. Schaefer, C. R. Schwalm, C. Williams, M. A. Arain, A. Barr, J. M. Chen, K. J. Davis, D. Dimitrov, T. W. Hilton, D. Y. Hollinger, E. Humphreys, B. Poulter, B. M. Raczka, A. D. Richardson, A. Sahoo, P. Thornton, R. Vargas, H. Verbeeck, R. Anderson, I. Baker, T. A. Black, P. Bolstad, J. Chen, P. S. Curtis, A. R. Desai, M. Dietze, D. Dragoni, C. Gough, R. F. Grant, L. Gu, A. Jain, C. Kucharik, B. Law, S. Liu, E. Lokipitiya, H. A. Margolis, R. Matamala, J. H. McCaughey, R. Monson, J. W. Munger, W. Oechel, C. Peng, D. T. Price, D. Ricciuto, W. J. Riley, N. Roulet, H. Tian, C. Tonitto, M. Torn, E. Weng, and X. Zhou, "A model-data comparison of gross primary productivity: Results from the North American Carbon Program site synthesis," *Journal of Geophysical Research: Biogeosciences*, vol. 117, no. G3, 2012. [Online]. Available: http://dx.doi.org/10.1029/2012JG001960

[19] S. Zaehle, B. E. Medlyn, M. G. De Kauwe, A. P. Walker, M. C. Dietze, T. Hickler, Y. Luo, Y.-P. Wang, B. El-Masri, P. Thornton, A. Jain, S. Wang, D. Warlind, E. Weng, W. Parton, C. M. Iversen, A. Gallet-Budynek, H. McCarthy, A. Finzi, P. J. Hanson, I. C. Prentice, R. Oren, and R. J. Norby, "Evaluation of 11 terrestrial carbon-nitrogen cycle models against observations from two temperate Free-Air CO_2 Enrichment studies," *New Phytologist*, 2014. [Online]. Available: http://dx.doi.org/10.1111/nph.12697

[20] B. Parhami, "SIMD machines: Do they have a significant future?" *SIGARCH Comput. Archit. News*, vol. 23, no. 4, pp. 19–22, Sep. 1995. [Online]. Available: http://doi.acm.org/10.1145/218864.218868

[21] T. Qian, A. Dai, K. E. Trenberth, and K. W. Oleson, "Simulation of global land surface conditions from 1948 to 2004. Part I: Forcing data and evaluations." *Journal of Hydrometeorology*, vol. 7, no. 5, pp. 953–975, 2006.

[22] Y. Wei, S. Liu, D. N. Huntzinger, A. M. Michalak, N. Viovy, W. M. Post, C. R. Schwalm, K. Schaefer, A. R. Jacobson, C. Lu, H. Tian, D. M. Ricciuto, R. B. Cook, J. Mao, and X. Shi, "The North American Carbon Program Multi-scale Synthesis and Terrestrial Model Intercomparison Project part 2: Environmental driver data," *Geoscientific Model Development Discussions*, vol. 6, no. 4, pp. 5375–5422, 2013. [Online]. Available: http://www.geosci-model-dev-discuss.net/6/5375/2013/

[23] K. E. Mitchell, D. Lohmann, P. R. Houser, E. F. Wood, J. C. Schaake, A. Robock, B. A. Cosgrove, J. Sheffield, Q. Duan, L. Luo, R. W. Higgins, R. T. Pinker, J. D. Tarpley, D. P. Lettenmaier, C. H. Marshall, J. K. Entin, M. Pan, W. Shi, V. Koren, J. Meng, B. H. Ramsay, and A. A. Bailey, "The multi-institution North American Land Data Assimilation System (NLDAS): Utilizing multiple GCIP products and partners in a continental distributed hydrological modeling system," *Journal of Geophysical Research: Atmospheres*, vol. 109, no. D7, 2004. [Online]. Available: http://dx.doi.org/10.1029/2003JD003823

[24] L. DeRose, B. Homer, and D. Johnson, "Detecting application load imbalance on high end massively parallel systems," in *Euro-Par 2007 Parallel Processing*, ser. Lecture Notes in Computer Science, A.-M. Kermarrec, L. Boug, and T. Priol, Eds. Springer Berlin Heidelberg, 2007, vol. 4641, pp. 150–159. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-74466-5_17

[25] A. Bhatele, K. Mohror, S. H. Langer, and K. E. Isaacs, "There goes the neighborhood: performance degradation due to nearby jobs," in *Proceedings of SC13: International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM, 2013, p. 41.

[26] L. V. Kale, G. Zheng, C. W. Lee, and S. Kumar, "Scaling applications to massively parallel machines using projections performance analysis tool," in *Future Generation Computer Systems Special Issue on: Large-Scale System Performance Modeling and Analysis*, vol. 22, no. 3, February 2006, pp. 347–358.

[27] J. Li, W. K. Liao, A. Choudhary, R. Ross, R. Thakur, W. Gropp, R. Latham, A. Siegel, B. Gallagher, and M. Zingale, "Parallel netCDF: A high-performance scientific I/O interface," *SC Conference*, vol. 0, p. 39, 2003.

[28] B. Austin, M. J. Cordery, H. J. Wasserman, and N. J. Wright, "Performance measurements of the NERSC Cray Cascade system," in *Cray User Group*, 2013. [Online]. Available: https://cug.org/proceedings/cug2013_proceedings/includes/files/pap156.pdf

[29] E. Totoni and L. V. Kale, "ACM SRC poster: optimizing all-to-all algorithm for PERCS network using simulation," in *Proceedings of the 2011 companion on High Performance Computing Networking, Storage and Analysis Companion*, ser. SC '11 Companion. ACM, 2011, pp. 123–124.

[30] NERSC. Edison file storage and I/O. [Online]. Available: http://www.nersc.gov/users/computational-systems/edison/file-storage-and-i-o/

[31] Message Passing Interface Forum, "MPI: A message-passing interface standard version 3.0," Tech. Rep., September 2012.