High Performance Computing Enabled Simulation of the Food-Water-Energy System

Simulation of Intensively Managed Landscapes

Harold E.B. Dennis
Indiana University
Bloomington, Indiana, U.S.A. 47401
hbrokaw@iu.edu

Adam S. Ward
Indiana University
Bloomington, Indiana 47401
adamward@indiana.edu

Tyler Balson
Indiana University
Bloomington, Indiana, U.S.A. 47401
tbalson@umail.iu.edu

Yuwei Li
Curacloud Corporation
Seattle, Washington, U.S.A. 98104
yuwei.lee03@gmail.com

Robert Henschel
Indiana University
Bloomington, Indiana, U.S.A. 47401
henschel@iu.edu

Shawn Slavin
Indiana University
Bloomington, Indiana, U.S.A. 47401
slavin@iu.edu

Stephen Simms
Indiana University
Bloomington, Indiana, U.S.A. 47401
ssimms@iu.edu

Holger Brunst
Technische Universitt Dresden,
Center for Information Serv. & HPC
Dresden, Germany 01062
holger.brunst@tu-dresden.de

ABSTRACT

Domain science experts are commonly limited by computational efficiency of their code and hardware resources available for execution of desired simulations. Here, we detail a collaboration between domain scientists focused on simulating an ensemble of climate and human management decisions to drive environmental (e.g., water quality) and economic (e.g., crop yield) outcomes. Briefly, the domain scientists developed a message passing interface to execute the formerly serial code across a number of processors, anticipating significant performance improvement by moving to a cluster computing environment from their desktop machines. The code is both too complex to efficiently re-code from scratch and has a shared codebase that must continue to function on desktop machines as well as the parallel implementation. However, inefficiencies in the code caused the LUSTRE filesystem to bottleneck performance for all users. The domain scientists collaborated with Indiana University's Science Applications and Performance Tuning and High Performance File System teams to address the unforeseen performance limitations. The non-linear process of testing software advances and hardware performance is a model of the failures and successes that can be anticipated in similar applications. Ultimately, through a series of iterative software and hardware advances the team worked collaboratively to increase performance of the code, cluster, and file system to enable more than 100-fold increases in performance. As a result, the domain science is able to assess ensembles of climate and human forcing on the model, and sensitivities of ecologically and economically important outcomes of intensively managed agricultural landscapes.

CCS CONCEPTS

•**Applied computing** →**Earth and atmospheric sciences;**

KEYWORDS

Agro-IBIS, agro-ecosystem, modeling, case study, benchmarking, mpi, hpc, vampir, scaling, performance, lustre, meta-data, filesystems, computer cluster, parallel computing

ACM Reference format:
Harold E.B. Dennis, Adam S. Ward, Tyler Balson, Yuwei Li, Robert Henschel, Shawn Slavin, Stephen Simms, and Holger Brunst. 2017. High Performance Computing Enabled Simulation of the Food-Water-Energy System. In *Proceedings of PEARC17, New Orleans, LA, USA, July 09-13, 2017,* 10 pages. DOI: 10.1145/3093338.3093381

1 INTRODUCTION

1.1 Use Case: Simulation of Intensively Managed Landscapes

The intensively managed landscapes of the agricultural Midwestern U.S. are among the world's most productive areas in terms of food and energy crops [8, 21]. This productivity is a result of nutrient-rich and water-holding soils, bolstered further by intense management of drainage, pests, diseases, and plant nutrients. Pressures on the Midwest's natural resources emerge from a number of sources: climate, water for biofuels [17], competing demands for land use, and urban populations. Tensions between crop production and the environmental impacts of agriculture are expected to grow as climate changes impact production potential [9] and

PEARC17, New Orleans, LA, USA

DOI: 10.1145/3093338.3093381

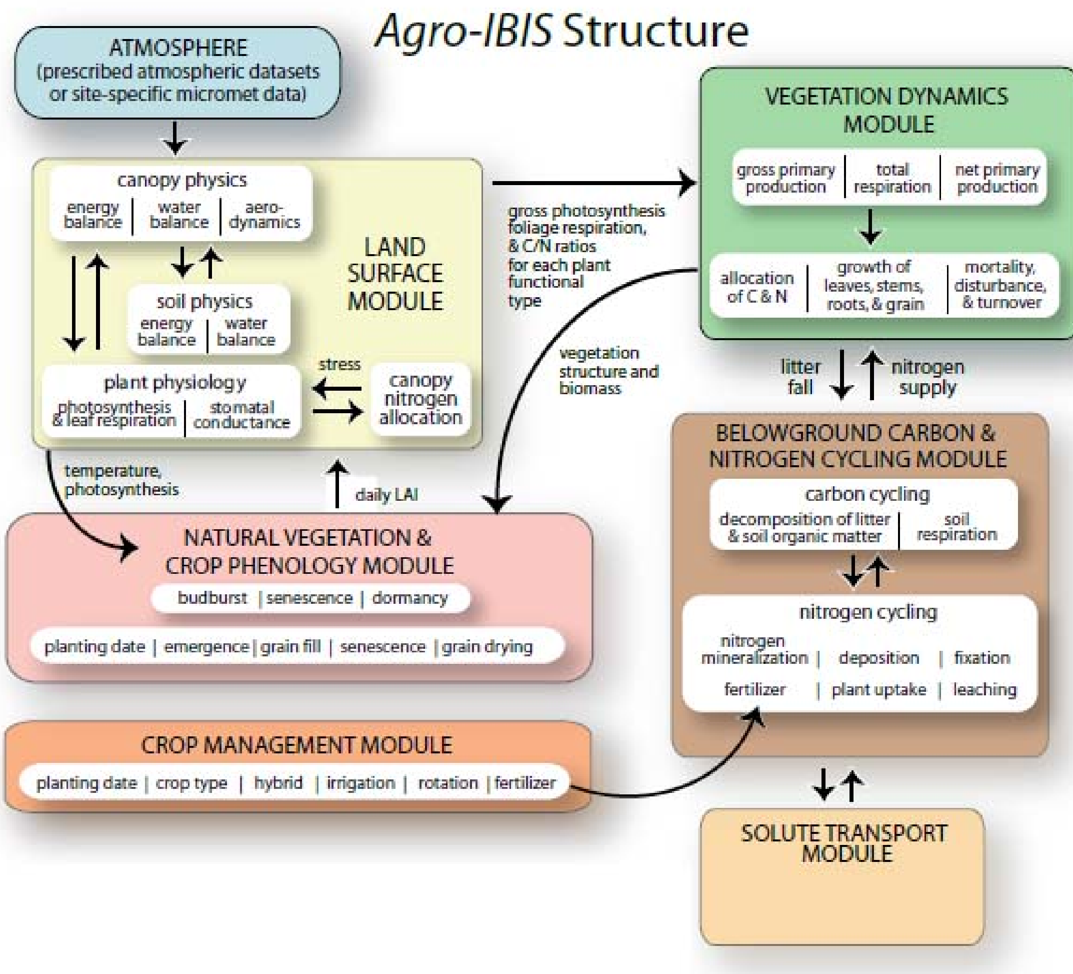


Figure 1: Conceptual model of the representation of earth surface and subsurface processes in Agro-IBIS including the simultaneous balance of energy, water, momentum, carbon, and nitrogen. The model takes key inputs of climate forcing and on-farm management (e.g., crop type, fertilizer timing and amount) and predicts a host of important outcomes including crop yield, soil health, water quality, and water quantity in Intensively Managed Landscapes. This figure reprinted from Fig. 1 in [13], ©American Meteorological Society. Used with permission.

stakeholders respond with litigation over downstream impacts of land management on water quality [26]. Changes that are already being experienced include warmer winters, longer growing seasons, increases in heavy rainfall, shifts in precipitation timing, and higher average streamflow [3, 23, 25, 34]. These changes in hydrological forcing will exacerbate the already large nutrient exports from the region [4, 29, 35]. Increasing soil erosion due to higher runoff [18, 19], changes in corn yield due to high temperatures [24], and production loss due to heavy rainfall [22] will all require adaptation by farmers.

Conflicts over agriculture's role in water quality are already prevalent, in part due to recognition of the region's contribution to high nutrient loads reaching local water supplies and the Gulf of Mexico [10]. Recent shifts in market dynamics have been attributed to the transition of degraded land in conservation programs back to more vulnerable annual systems [15]. Transitioning to perennial bioenergy sources, for example, can help mitigate nitrogen-related environmental impacts while providing increased soil health and reduced runoff and soil erosion [5]. However, to avoid unexpected consequences from such transitions, studying the integrated biological, physical, chemical and human impacts is critical. Because changing social and economic factors can dominate on-farm decision-making regarding what to plant, it is important that an analysis of land management scenarios explicitly incorporate dynamic (i.e., non-stationary) human behavior.

The suite of existing tools to simulate the interactions between climate and on-farm management decisions are commonly referred to as "agro-ecosystem models", typically modified versions of Earth System Models. One state-of-the-science agro-ecosystem model is Agro-IBIS [13], which has been widely validated in the Midwestern U.S. and used to forecast a host of management scenarios related to climate, energy crops, and crop rotation choices [2, 6, 14, 16, 28, 30–33]. Agro-IBIS requires inputs of dynamic forcing by natural systems (e.g., climate, weather), on-farm decisions (e.g., crop type, fertilizer application, conservation practices), and landscape properties (e.g., soil type, topography), making predictions of a host of state variables and fluxes related to carbon, nitrogen, water, and energy balances. Notable outcomes include measures of crop yield, soil quality, stormwater runoff, and nutrient losses. Taken together these outputs provide a suite of indicators that are important to Midwestern land managers and allow them to evaluate their current decisions and explore a host of possible future scenarios.

With the agro-ecosystem model in-hand, one key challenge remains. The model itself originated as v2 of the Integrated BioSphere Simulator, IBIS [7]. The code was written in Fortran-90 and designed to be run on a single processor on a stand-alone machine. As the code evolved into the modern Agro-IBIS, two notable and related developments were realized. First, a user-group formed that maintains the code using a SubVersioning network to facilitate sharing of modifications and updates to the code. Any modifications to the code must be compatible with the single-processor Fortran-90 based code to maintain usability across the community. The desire to maintain such usability for the network and lack of computing expertise resulted in the code failing to take advantage of newer computer architectures which contain multiple processors and considerably more memory than those for which the code was originally designed. Related to this challenge, code modifications and updates are conducted by domain scientists who have little expertise in computing optimization. As a result, the implementation of the model prior to this project was limited by computational power. Among the user group, having a dedicated PC to run simulations one-at-a-time on a single core is common. Furthermore, most domain scientists accept the long timescales for computation (e.g., one week or more for a simulation of 60 years across the Mississippi River Basin). This configuration necessarily limits the sheer number of runs that can be completed, and thus the ability to explore complex scenarios that systematically vary different climate and on-farm management inputs.

In summary, the user community is currently limited by the implementation of their code on a single processor. The community has aspirations to apply the model in two directions. First, the ability to forecast ensembles of climates and management decisions will inform sustainable resource management and allow the calculation of uncertainty envelopes around forecasts. Next, the group has discussed near real-time forecasting of water quality that would require rapid throughput. In the remainder of this manuscript we document the process of modifying the code base to run on Indiana University's High Performance Computing infrastructure, requiring notable advances in hardware and software.

2 BACKGROUND

Our team includes Indiana University (IU) scholars, Ward, Li, and Balson (Ward's group) and members of IU's HPC Research Applications group (Henschel's group). Ward's group studies the transport and fate of solutes through the landscape, with a particular emphasis in this project on agricultural systems and nutrient pollution. Henschel's group is dedicated to serving the IU research community, facilitating their use of IU's Advanced Cyberinfrastructure facilities. In addition to members of his Science Community Tools division, Henschel invited Simms to bring members of his High Performance File System group (HPFS) to participate. Ward's group had developed mpi4ibis in order to run Agro-IBIS on multiple processors and accelerate their ability to explore a wider range of scenarios and input configurations. Although not the original reason why Ward contacted Henschel's group for assistance, while scaling up the number of parallel processes, it was discovered that running mpi4ibis on larger numbers of processors had a significant negative impact on the functioning of our computational file system, Data Capacitor II.

2.1 IU's Advanced Cyberinfrastructure

Big Red II [11] and Big Red II Plus are two of the prominent HPC systems within IU's Advanced Cyberinfrastructure. Big Red II is a shared resources for researchers, faculty, staff and students at IU. Big Red II Plus is Indiana University's newest high performance computing system dedicated to large projects and in particular to projects funded through Indiana University's grand challenge initiative. It is a Cray XC30 comprised of 552 compute nodes, each of which contains two Intel Xeon E5-2697 v2 12-core processors and 64 GB of RAM. Big Red II is a Cray XE6/XK7 supercomputer and is IU's primary HPC system designed for parallel computing. It is comprised of a hybrid architecture containing 344 CPU-only compute nodes and 676 GPU/CPU compute nodes accounting for 1020 total nodes. Both systems contain I/O nodes that bridge the Cray internal interconnect into the Infiniband fabric for fast access to the Data Capacitor II (DC2) [12].

The DC2 is the primary networked scratch and project storage platform for IU's research computing systems. It is a 5.3 PB, Lustre-based file system. The system is composed of one active meta-data server (MDS) and sixteen object storage servers (OSSs), which manage I/O to 252 object storage targets (OSTs), each of which is a pool of redundant hard drive storage. The system is also supported by an arrangement of Lustre networking (or LNET) routers, which are specially configured servers designed to route data between DC2 and the Cray Aries and Gemini interconnects, as well as 10 Gb/s Ethernet from IU's Karst and Mason clusters. The DC2 interconnect is based on 56 Gb/s FDR Infiniband, and it connects to the Lustre routers via a 108-port Mellanox SX6506 FDR Infiniband switch.

2.2 MPI implementation

Prior to engaging with Henschel's group, Li and Ward had developed mpi4ibis, a multiple processor implementation (MPI) for the Agro-IBIS code. Agro-IBIS calculations work by decomposing the domain into a grid of cells to represent the landscape, and then applying calculations to each individual cell. To maintain the ability to integrate the code with the SubVersion network, the MPI was implemented in C++ as a ”wrapper” that would control the execution of the program without alteration of the executable file. Briefly, the steps for the MPI:

(1) Decompose the domain into a series of sub-domains;
(2) Build the necessary input files for each sub-domain;
(3) Execute each sub-domain on a different processor using the Agro-IBIS executable from the Fortran code; and
(4) Post-processing to stitch the files from each processor into a single set of outputs

This strategy allowed the simulation to be divided amongst as many processors as desired. Although the added steps of pre-processing to define the domains and post-processing to combine results were required, run-times were significantly reduced compared to those possible on a single office PC.

2.3 Initial Consultation

When mpi4ibis was first brought to Henschel’s group, it was able to compute much larger areas in a shorter time than had been possible using a single serial Agro-IBIS process. Ward’s team had calculated the time that they thought would be necessary to complete their analysis and was requesting exclusive or priority use of the system in order to be able to complete the work. Importantly, the request would have run several instances using hundreds of processors each in parallel to achieve the desired results. Even with the considerable advances in speed, Ward needed higher throughput than what could be achieved by submitting jobs in the usual way.

During our team’s initial consultation meeting it was decided to do some benchmarking and some performance analysis of mpi4ibis. Considering the lack of prior analysis, it was considered likely that we could increase the performance of the software and reduce the resources required to perform the research. The performance analysis was not straight forward, because from the point of view of mpi4ibis, the Agro-IBIS subprocesses are black boxes. It was necessary to analyze Agro-IBIS separately from mpi4ibis.

2.4 Unanticipated Problem

Concurrent with our analyses, Ward’s group ran multi-node mpi4ibis jobs to test scaling and performance as well as to test a variety of changes in the parameters space. These test runs were taking considerably longer than anticipated. At the same time, other users of the DC2 file system were reporting slow system response. Even basic file system operations such as ”ls” could take unreasonably long to complete. It was determined that the system slowdown was happening whenever Ward’s group had multiple jobs running. Although there were environmental factors which could have been causing some of the response problems, whenever Ward’s group would complete or halt all of their runs, the system would return to its typical behavior. As Ward increased the number and size of his tests, running mpi4ibis using more Agro-IBIS sub-processes 256, 512, 1024, etc., the detrimental affect on the DC2 Lustre file system became clear. The file system would slow down to a crawl when a number of Ward’s jobs were running on the system. The ultimate result was that system performance declined for all users including Ward’s group.

3 ANALYSES

3.1 Analysis Procedures

The code builds easily on our Cray System using the Intel compiler and NetCDF libraries. As described above, the Agro-IBIS code is serial and was written in Fortran. The MPI C++ wrapper assigns the sub-domains of the problem to the various MPI tasks which then run Agro-IBIS on that sub-domain through a system call, as a sub-process. Each Agro-IBIS sub-process creates its own set of output files, which then must be post-processed to stitch them together into a single set of output files.

Our performance analysis of the application was done using Score-P [27], Vampir [20], and Allinea Map [1]. We had to create a separate instrumentation to analyze mpi4ibis than the one created to analyze Agro-IBIS. Benchmarking of the code was performed on maintenance days, prior to opening up the HPC systems, so that we could observe performance on a quiescent system. This allowed our relatively short test run times to be less affected by external factors. A short demonstration problem was split into 16 parts. Each Agro-IBIS run in these tests was set up to compute a problem corresponding to one of the 16 parts, guaranteeing that they are all computing problems of similar size, as they do during production. Various environmental parameters were then changed, including increasing the number of simultaneously running Agro-IBIS processes.

3.2 Performance Analysis

Figure 2 shows results from analyzing a simple demonstration problem that completes in a few minutes. The computational section of the code is shown as evenly distributed over the utilized processes. Post-processing is shown to be dominated by the processing of a single file by a single processor. At the time of this analysis, computational times for production runs were longer than post-processing times. So in the short term we focused on Agro-IBIS code analysis, leaving improving the post-processing performance to later.

Vampir traces of Agro-IBIS showed that there was a considerable amount of I/O that was happening, even within primarily computational sections of the code. Figure 3 is analysis of a single Agro-IBIS run corresponding to one of the green sections in figure 2. The yellow and purple are read and write sections of the code. These dominate the time consumed. Our Allinea [1] performance report for Agro-IBIS indicated that 30.3% of the time is spent in computation and 69.7% in I/O. Within the I/O, about twice as much time is spent in writes than in reads. Note the two insets in figure 3 showing small sections of the trace in more detail. Small reads and writes occur regularly and in great numbers throughout the course of the program.

3.3 Environmental Analysis

Benchmark tests were performed using a small problem set, without the post-processing. We did this to determine if we could change system parameters in ways that would improve the operability of the code on our system. We experimented with different NetCDF buffer sizes. We tried to use Lustre striping in ways that might

[1] Thanks to Le Mai Nguyen Weakly who performed the Allinea work.

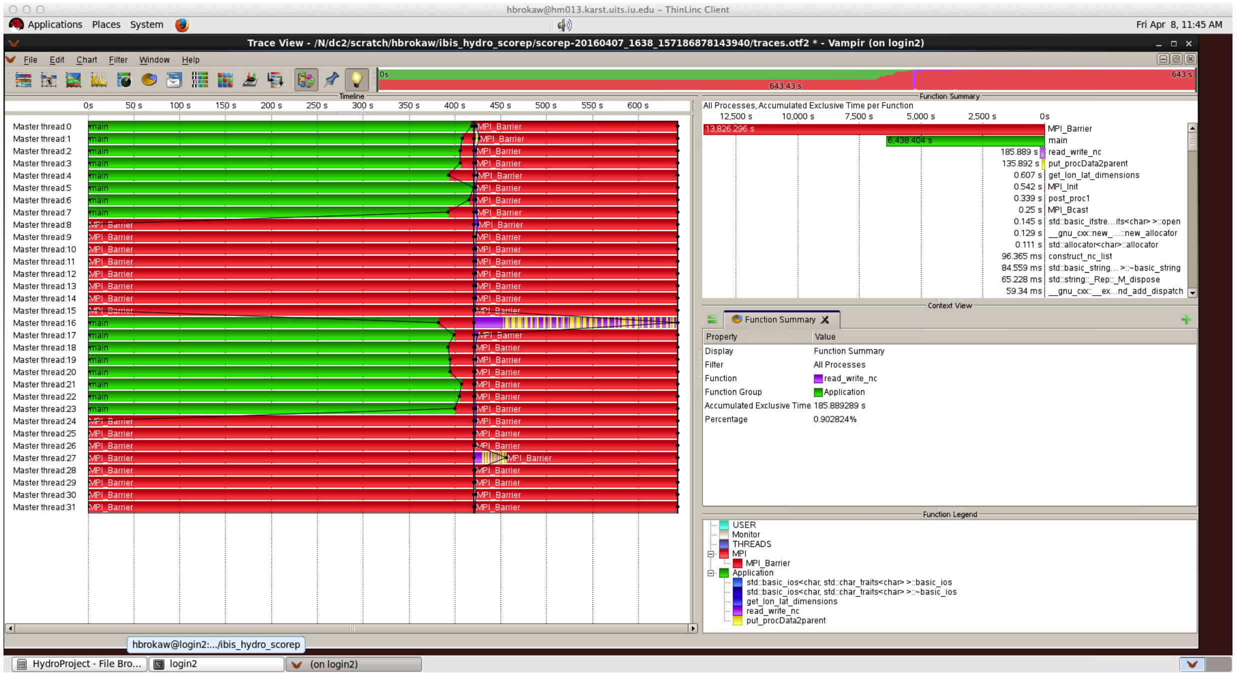


Figure 2: Vampir trace of a short demonstration run of mpi4ibis. Only half of the processes are utilized in the computational phase. The green sections labeled "main" are the system calls to AgroIBIS. The red sections are where processes are waiting at MPI barriers waiting for all of the other processes to reach the barrier. The post processing phase begins after the MPI Barrier in the middle of the figure. All processors are utilized in the post-processing phase, however the post processing is distributed on a per file basis, so most of the processors are waiting for the post-processing of one large file to complete.

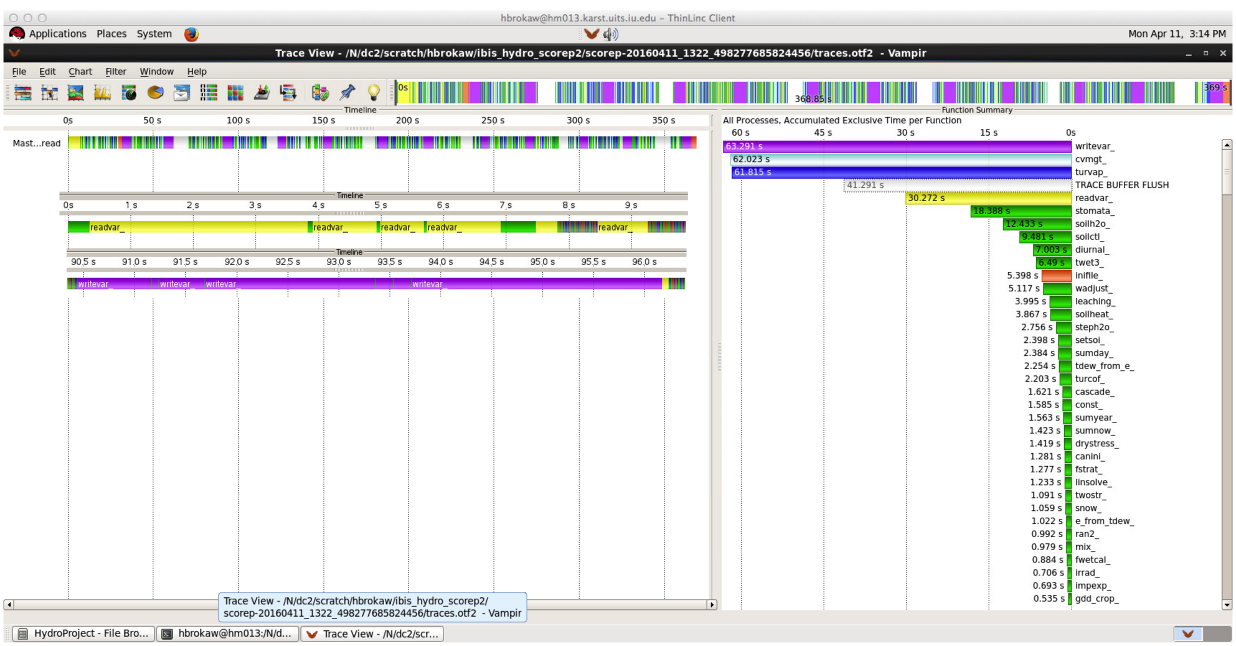


Figure 3: Vampir trace of one Agro-IBIS process corresponding to one of the Agro-IBIS process runs in figure 2. Two insets pasted in below the trace of the code show a zoom into the initial read section (0-10 secs) and a write section (90.5-96.0 secs)

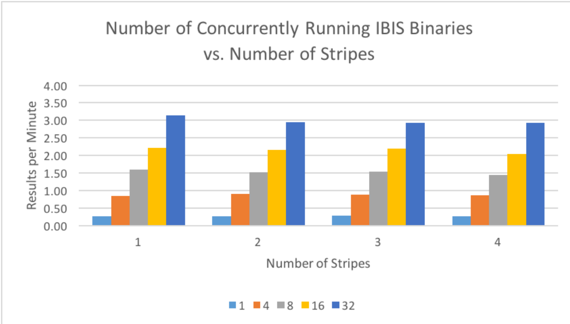
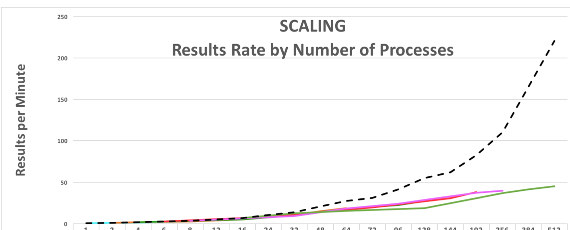


Figure 4: Increasing the number of stripes used by the directories where the data is read and written is associated with a marginal decrease in performance.

increase throughput of data and diminish the impact of the code on the file system (figure 4). We also tried using a dedicated Lustre OST pool for this application, to direct the I/O to disks with more space and slightly less background activity. However, we were not able to reduce the impact of the code on the system or increase its speed in a significant way through any of these measures.



	1	2	4	6	8	12	16	24	32	48	64	72	96	128	144	192	256	384	512
1 node	0.43	0.92	1.42		2.76	4.26	5.89	8.14	10.97										
2 nodes		0.81	1.68		2.68		5.27	8.04	10.49	15.08	18.55								
4 nodes			1.66		3.66		5.00		10.11	14.05	17.94		22.33	27.43					
6 nodes				2.45		5.41		7.87		14.05		19.55	23.23		30.64	38.27			
8 nodes					3.33		6.40		9.10		18.37		24.10	28.44		37.28	39.90		
16 nodes							6.76		12.47		15.55			18.69		30.72	37.01	41.51	45.31
Ideal	0.43	0.86	1.72	2.58	3.44	5.16	6.88	10.32	13.76	20.64	27.52	30.96	41.28	55.04	61.92	82.56	110.08	165.12	220.16

Figure 5: Scaling of results per minute. Each Agro-IBIS process is solving a similar size problem. Because individual processors do not communicate during execution, scaling should be close to ideal. The divergence from ideal scaling is significant considering the lack of interprocess overhead. (Slope changes in the ideal line are due to the non-linear nature of the axis scale.)

Scaling of the code computationally should be close to ideal, since there is no inter-process communication among the Agro-IBIS processes. Consequently, there should be very little run-time increase as the number of concurrent processes increases and the "Results per Minute" should increase proportional to the number of processes. However, the scaling diverges considerably from the ideal (figure 5). What we saw was that as the the number of Agro-IBIS processes running increases, at a certain point, in terms of number of processes, the processes start slowing down considerably. This is also shown in figure 6, which shows dramatic increases in time as the number of processes increase. In ideal scaling the time would stay constant as the number of processes increases.

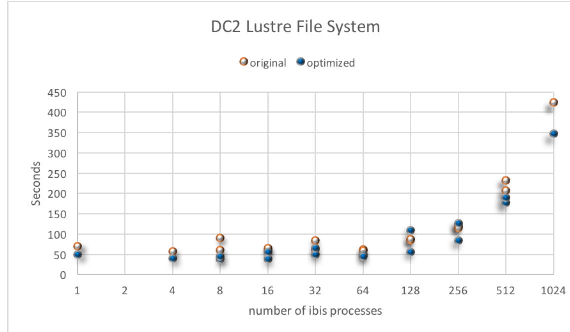


Figure 6: Computational time as a function of the number of processors running Agro-IBIS simulations with mpi4ibis code as originally constructed compared to the code as we optimized it after our analyses. Reads and writes are occurring on DC2. Each Agro-IBIS process is solving a similar size problem. Ideally the line in this graph would be relatively flat.

When the negative impact of mpi4ibis on DC2 was discovered, we did some additional tests to measure the effect of the code running on the file system. With 512 Agro-IBIS processes running on an otherwise quiescent system, I/O operations per second (IOPS) peaked at around 25,000 and write I/O peaked at 24 GB per second. This peak I/O is about half of the demonstrated peak throughput. The number of of IOPS is an order of magnitude above typical sustained values and approximately 2–3 times the typical short-term burst values seen during normal load on DC2 from all systems that rely on its storage.

3.4 Interpretation

The bottleneck to performance enhancement was the I/O. Our analysis is that this is because of competition for I/O resources and in particular competition for access to the meta-data server. The I/O pattern of the Agro-IBIS code was not sufficiently adapted to scale with the parallel execution under the management of a MPI C++ management wrapper. The multiplication of a large number of IOPS by the parallel runtime led to an unacceptable slow-down of DC2. One analogy that Slavin used to describe what mpi4ibis was doing is that of filling teacups with firehoses in order to fill a swimming pool. Optimization of the I/O strategies in the Fortran code of Agro-IBIS was clearly needed. That work is described below, and falls into two main categories: read optimizations and localizing write operations to the compute nodes.

An important consideration on our system is that the Cray-supported, Lustre-enhanced Linux kernel available to us is Lustre version 2.5.3. However, the DC2 filesystem, at the time of this research, was running Lustre version 2.1, and therefore supported only one meta-data server. As well, the version of the client kernel on BRII also lacks support for multiple meta-data servers. This means that any increase in meta-data activity (file opens, closes, stats, etc.) will not scale with the increase in the number of processors used to run the parallel Agro-IBIS simulations. Our optimizations attempt to resolve this to some degree, but a complete parallel I/O optimization would require major rewriting of the core Agro-IBIS code, which is undesired at this time due to the shared code base.

When the Agro-IBIS code was initially written, memory was an important constraint on computation. In order to reduce memory use, or perhaps as well for simplicity in the code, Agro-IBIS opens the file before, then closes the file after, each I/O operation. This happens piecemeal for each read or write call, rather than doing bulk reads of the data, then accessing it through memory, or accumulating data in memory, and then writing it out in chunks (figure 7). Another factor to note, but which only became apparent to us in post analysis, is that a single NetCDF I/O operation can result in multiple POSIX I/O operations because of the multidimensional nature of NetCDF I/O. This can result, for example, in a call to nc_get_vara_float() requesting 2K actually reading in 100K during the operation.

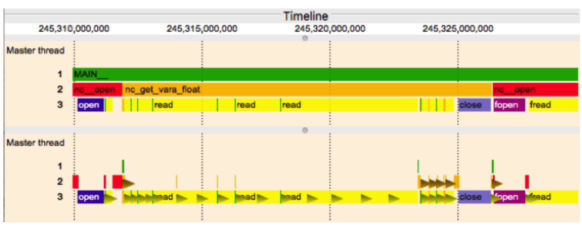


Figure 7: High level detail of nc_get_vara_float shows calls to open and close before and after. This pattern occurs with every single read and write in Agro-IBIS.

4 OPTIMIZATIONS

While rewriting Agro-IBIS to be more efficient with I/O could have been seen as a useful proposition, it was ruled out, due to the community codebase. Rewritten code would have to prove that it produced the same results and it was not considered within the scope of this project to engage in that task. Additionally, the requirement that the core program maintain separability as a stand alone serial program ruled out certain types of I/O optimizations.

4.1 File Reading Optimization

The Agro-IBIS Fortran code was initially modified by Ward's team to run as independent, parallel processes managed by an MPI management code, mpi4ibis, written in C++. The I/O functionality was modified to utilize the standard NetCDF-4 data format. As noted above, the read and write functionality was maintained as atomic operations involving opening and closing of files on each read and write. Slavin implemented a simple modification of the code to manage I/O streams and minimize file opens and closes. This provided a modest performance gain. More significantly, this modification reduced the impact of numerous Agro-IBIS processes running concurrently on the file system. This is because it considerably reduced the demands on the meta-data server.

Additional experiments using different user-defined buffer cache sizes to minimize file read operations, provided us a minimal additional optimization for the portion of the runtime involved in reading historical records to condition each Agro-IBIS timestep.

4.2 RAM Disk Optimization

Nodes on BRII have 32GB per CPU, half of which is available to use as a ram disk for the whole node, via /tmp. Dennis proposed that rather than going to disk for every write operation, we stage our computations out of /tmp, keeping the writes in ram, then transferring the output files in a chunk, when computation was finished.

This sped up the code considerably, since the I/O no longer had to wait for the Lustre file system in order to write out each small batch of data and no longer had to contend for access to the metadata server. As well, writing to local memory is much faster than writing to an external hard drive system. There was the added time of transferring the data in a chunk to the hard drive system, but the Lustre file system is much better at writing large chunks than it is writing many small files.

During testing of this modification, it was determined that we would need to write out the data for each year, in order to avoid filling the memory in /tmp during multi-year runs. The change also created a limit to how many Agro-IBIS processes we could have running on each node, depending on the problem size, without overflowing the available RAM in /tmp. In practice, it was easiest to opt for a lower number in order to avoid Out of Memory errors, so 8 processes per CPU was most commonly used during production runs.

4.3 Post Processing Optimization

The increase in the throughput of the Agro-IBIS processes meant that the post-processing now took longer to complete than the Agro-IBIS computations. Through code inspection, Dennis discovered that the mpi4ibis code replicated the Agro-IBIS pattern of opening and closing the file for each value read. The open/close operations were moved out of the enclosing loops and file id's saved in an array. While we did not directly measure the results of this change, Ward reported a tenfold increase in the speed of the post-processing. This balanced the workflow so that post-processing was no longer holding up the computation.

4.4 Implementation of an SSD-based File System

During the period in which this optimization work was being done, the DC2 file system was running very close to its capacity, just ahead of a major upgrade of 1.5 PB. In an attempt to give the Agro-IBIS work a boost in performance, and to move the taxing application off the main, DC2 file system, an experimental, SSD-based, Lustre file system was set up to support Ward's group's work. The experimental system is known as DCRAM.

The DCRAM file system is composed of eight Lustre servers, two for meta-data (MDS nodes) and six to manage object storage targets (OSS nodes). The DCRAM filesystem supports twelve object storage targets (OSTs), each of which is a RAID-0 (striped and concatenated) array of four 800 GB Intel enterprise MLC-based SSD drives. The formatted file system allows for up to twelve parallel write streams at a time to files which may be striped across OSTs, and via an Infiniband interconnect of 40 Gb/s connections to the BRII system. The dual MDS nodes allow for optional striping of meta-data, also

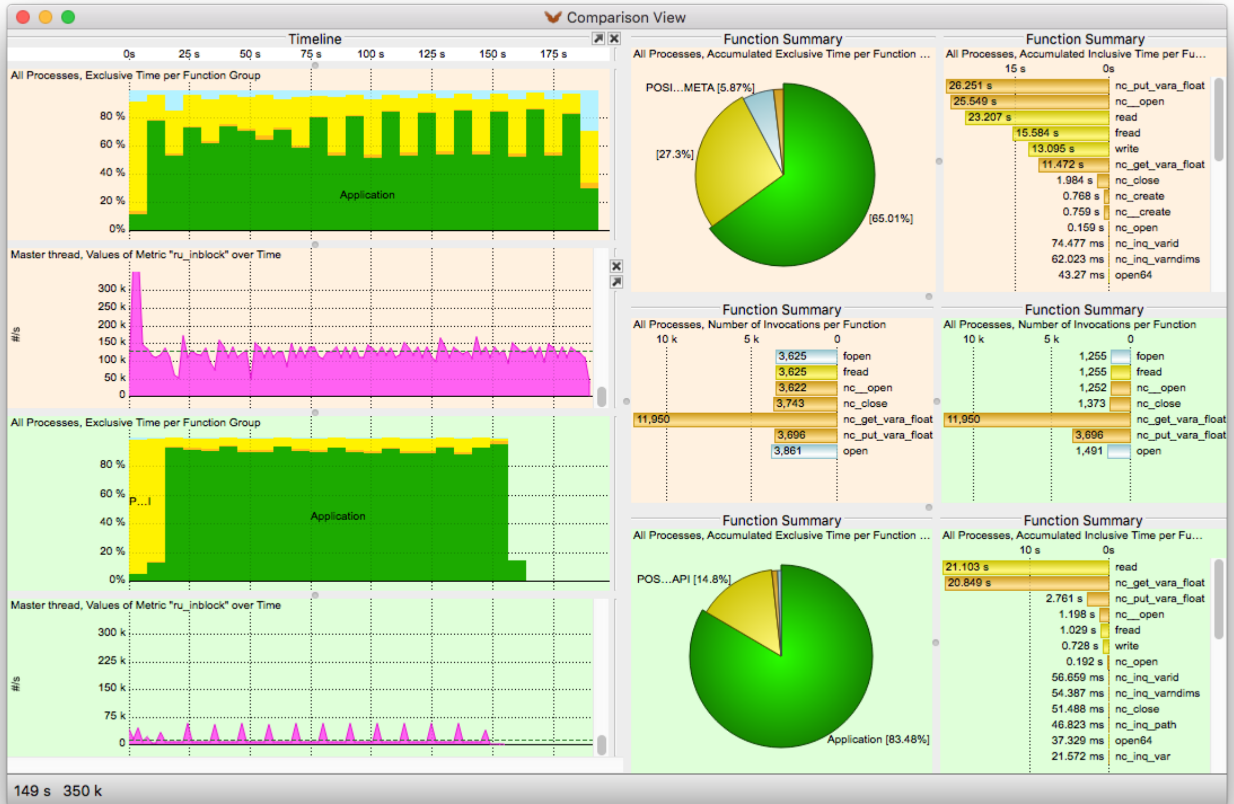


Figure 8: Vampir screen shot depicting a single instance of the Agro-IBIS code out of 16. Progress over time (about 180 seconds) is depicted on the left while aggregated stats are depicted on the right. The background color identifies the non-tuned (orange) and tuned (green) results.

via 40 Gb/s Infiniband connections to the 4-SSD RAID-0 arrays that comprised the MDT storage.

4.5 Comparison Analysis

We studied the performance of the code with the Vampir performance visualizer [20]. The respective performance data was recorded with the Score-P performance monitor [27]. Score-P was configured to record detailed event data for NetCDF calls, user functions and rusage resource counters. Data was recorded for both the Agro-IBIS MPI-wrapper, mpi4ibis, and the actual Agro-IBIS compute kernel.

The latter is discussed here primarily due to its high I/O demands. Detailed performance tests were performed on a 1 node/16 core configuration of our BRII system for simplicity reasons. The solver ran for about 180 seconds. At this scale performance issues start becoming observable. Larger performance tests were evaluated with absolute timings and system performance stats to prove what has been observed and tuned at a smaller scale.

Figure 8 illustrates the impact of our code improvements graphically. The left portion of this figure illustrates the behaviour of the code over time. Color coding is used to identify the different activities of the code. Yellow identifies POSIX I/O and orange is used for NetCDF. I/O read activity is depicted as graph in pink showing I/O blocks read per second. On the right hand side aggregated invocation numbers and times are depicted for the dominating POSIX and NetCDF calls of our code. The orange background color identifies the original version of the code whereas the green background color represents results for the tuned code version. A single representative instance of the Agro-IBIS code out of 16 has been used for analysis.

The following effects can be observed: Per core read input bandwidth has been brought down to 10% of the original demand (see pink graphs). Meta-data handling due to calls to nc_open() and nc_close() has been reduced to a third (see table in the middle row to the right). The overall time spent in nc_put_vara_float() was reduced by a factor of ten due to the ram optimization mentioned earlier.

5 SCALABILITY ANALYSIS

Ward's recent work even using the optimized code was constrained more by file system response than by availability of computational resources. The post-processing steps also slowed the ability to move data into long term storage to make room for more data runs. In order to maximize Ward's ability to complete his work, he was given access to DCRAM to use for I/O. Without additional optimization, the scale of Ward's problem domain requires the highest throughput file system that we can currently provide.

Given access to dedicated resources, such as BigRed II+ along with the DCRAM file system, current optimizations are seen to degrade file system response on DCRAM (figure 9) in a way similar to DC2 (figure 6). Although throughput is faster, system degradation still curves up in a way that limits the parallelization of the code to even larger numbers of nodes (more concurrent Agro-IBIS subprocesses).

Further scaling can be achieved through optimization of the post-processing, or through elimination of it, via the use of parallel NetCDF to transfer the data from sub-domain files in local RAM, to the appropriate final destination files on the file system, rather than transferring the sub-domain files to the file system. That could additionally reduce meta-file operations, reducing impact on the file system as a whole. The more we can reduce I/O contention, the flatter we can make the above curves, the larger the problem domains that will be able to be approached.

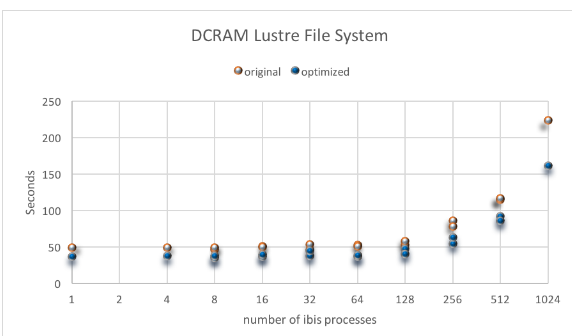


Figure 9: Computational timing as a function of the number of processors running Agro-IBIS using original mpi4ibis code vs. optimized code on DCRAM. Compare timings with figure 6.

6 CONCLUSION

In this case study, we demonstrate the iterative development and implementation of a solution to adapt a shared agro-ecosystem code to run in a HPC environment. The use-case was motivated by a model user community that was limited in their ability to conduct ensembles of simulations by extremely long run-times using code that was not programmed to execute in parallel.

First, we made only minor modifications to the source code itself, which consisted only of implementing best practices using NetCDF files. This was important to retain the portability of the code amongst the heterogeneous user base managed via the SubVersion network. We iteratively developed an MPI application in C++ to manage the Fortran-90 software, enabling scaling across many processors. Initial implementation of the MPI did not scale because the intensive I/O demands on the Lustre file system. As a result, modification of I/O timing and the use of on-node storage to aggregate outputs before storage on the file system were used to provide a scalable solution. At present, the domain scientists are conducting multiple projects that include hundreds of model simulations under a range of climate and management scenarios. This scale of simulation was not possible prior to our project. Future directions may include using parallel NetCDF commands to eliminate the post-processing step from the workflow, and implementation of the code for near-, mid-, and long-term forecasts of water quality and crop yield.

ACKNOWLEDGMENTS

We would like to acknowledge the work and contributions of Le Mai Nguyen Weakly who participated in this project as a part of Henschel's group during its early stages.

This research was supported in part by Lilly Endowment, Inc., through its support for the Indiana University Pervasive Technology Institute, and in part by the Indiana METACyt Initiative. The Indiana METACyt Initiative at IU is also supported in part by Lilly Endowment, Inc. This material is also based upon work supported by the National Science Foundation under Grant Nos. CNS-0521433, EAR-1331906, and EAR-1505309. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the author(s), and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Allinea Software. *Allinea MAP - C/C++ profiler and Fortran profiler for high performance Linux code.* https://www.allinea.com/products/map accessed on Jun 5, 2017.
[2] Justin E. Bagley, Jesse Miller, and Carl J. Bernacchi. 2015. Biophysical impacts of climate-smart agriculture in the Midwest United States. *Plant, cell & environment* 38, 9 (2015), 1913–1930.
[3] Shuwei Dai, Martha D. Shulski, Kenneth G. Hubbard, and Eugene S. Takle. 2016. A spatiotemporal analysis of Midwest US temperature and precipitation trends during the growing season from 1980 to 2013. *International Journal of Climatology* 36, 1 (2016), 517–525. https://doi.org/10.1002/joc.4354
[4] Caroline A. Davis, Adam S. Ward, Amy J. Burgin, Terrance D. Loecke, Diego A. Riveros-Iregui, Douglas J. Schnoebelen, Craig L. Just, Steven A. Thomas, Larry J. Weber, and Martin A. St Clair. 2014. Antecedent moisture controls on stream nitrate flux in an agricultural watershed. *Journal of environmental quality* 43, 4 (2014), 1494–1503.
[5] Sarah C. Davis, William J. Parton, Stephen J. Del Grosso, Cindy Keough, Ernest Marx, Paul R. Adler, and Evan H. DeLucia. 2012. Impact of second-generation biofuel agriculture on greenhouse-gas emissions in the corn-growing regions of the US. *Frontiers in Ecology and the Environment* 10, 2 (2012), 69–74.
[6] Simon D. Donner and Christopher J. Kucharik. 2008. Corn-based ethanol production compromises goal of reducing nitrogen export by the Mississippi River. *Proceedings of the National Academy of Sciences* 105, 11 (2008), 4513–4518.
[7] Jonathan A. Foley, I. Colin Prentice, Navin Ramankutty, Samuel Levis, David Pollard, Steven Sitch, and Alex Haxeltine. 1996. An integrated biosphere model of land surface processes, terrestrial carbon balance, and vegetation dynamics. *Global Biogeochemical Cycles* 10, 4 (1996), 603–628. https://doi.org/10.1029/96GB02692
[8] L. Guanter, Y. Zhang, M. Jung, J. Joiner, M. Voigt, J.A. Berry, and M.S. Moran. 2014. Global and time-resolved monitoring of crop photosynthesis with chlorophyl fluorescence. *Proceedings of the National Academy of Sciences* 111, 14 (April 2014), E1327–E1333. https://doi.org/10.1029/96GB02692
[9] J. Hatfield, C. Swanston, M. Janowiak, R. Steele, J. Hempel, J. Bochicchio, W. Hall, M. Cole, S. Hestvik, and J. Whitaker. 2015. Midwest and Northern Forests Regional Climate Hub Assessment of Climate Change Vulnerability and Adaptation and Mitigation Strategies, edited by T. *United States Department of Agriculture* (2015).

[10] Hypoxia Task Force. 2015. Mississippi River/Gulf of Mexico Hypoxia Task Force 2015 Report to Congress, 98 pp. (August 2015).

[11] Indiana University. *Big Red II at Indiana University.* https://kb.iu.edu/d/bcqt accessed on March 13, 2017.

[12] Indiana University. *The Data Capacitor II and DC-WAN high-speed file systems at Indiana University.* https://kb.iu.edu/d/avvh accessed on March 13, 2017.

[13] Christopher J. Kucharik and Kristofor R. Brye. 2003. Integrated BIosphere Simulator (IBIS) yield and nitrate loss predictions for Wisconsin maize receiving varied amounts of nitrogen fertilizer. *Journal of environmental quality* 32, 1 (2003), 247–268.

[14] Christopher J. Kucharik, Andy VanLoocke, John D. Lenters, and Melissa M. Motew. 2013. Miscanthus establishment and overwintering in the Midwest USA: a regional modeling study of crop residue management on critical minimum soil temperatures. *PloS one* 8, 7 (2013), e68847.

[15] Philip E. Morefield, Stephen D. LeDuc, Christopher M. Clark, and Richard Iovanna. 2016. Grasslands, wetlands, and agriculture: the fate of land expiring from the Conservation Reserve Program in the Midwestern United States. *Environmental Research Letters* 11, 9 (2016), 094005.

[16] Melissa M. Motew and Christopher J. Kucharik. 2013. Climate-induced changes in biome distribution, NPP, and hydrology in the Upper Midwest US: A case study for potential vegetation. *Journal of Geophysical Research: Biogeosciences* 118, 1 (2013), 248–264. https://doi.org/10.1002/jgrg.20025

[17] National Research Council et al. 2008. *Water implications of biofuels production in the United States.* National Academies Press. pp. 76.

[18] National Research Council et al. 2011. *Adapting to the impacts of climate change.* National Academies Press.

[19] Monte R. O'Neal, M.A. Nearing, Roel C. Vining, Jane Southworth, and Rebecca A. Pfeifer. 2005. Climate change impacts on soil erosion in Midwest United States with changes in crop management. *Catena* 61, 2 (2005), 165–184.

[20] Paratools. *Vampir.* http://www.paratools.com/Vampir accessed on March 13, 2017.

[21] Deepak K. Ray, Nathaniel D. Mueller, Paul C. West, and Jonathan A. Foley. 2013. Yield trends are insufficient to double global crop production by 2050. *PloS one* 8, 6 (June 2013), e66428. https://doi.org/10.1371/journal.pone.0066428

[22] Cynthia Rosenzweig, Francesco N. Tubiello, Richard Goldberg, Evan Mills, and Janine Bloomfield. 2002. Increased crop damage in the US from excess precipitation under climate change. *Global Environmental Change* 12, 3 (2002), 197–202. https://doi.org/10.1016/S0959-3780(02)00008-0

[23] Karen R. Ryberg, Wei Lin, and Aldo V. Vecchia. 2012. Impact of climate variability on runoff in the North-Central United States. *Journal of Hydrologic Engineering* 19, 1 (2012), 148–158.

[24] Jane Southworth, J.C. Randolph, M. Habeck, O.C. Doering, R.A. Pfeifer, D. Gangadhar Rao, and J.J. Johnston. 2000. Consequences of future climate change and changing climate variability on maize yields in the midwestern United States. *Agriculture, Ecosystems & Environment* 82, 1 (2000), 139–158. https://doi.org/10.1016/SOl67-8809(00)00223-1

[25] Kayla J. Steffens and K.J. Franz. 2012. Late 20th-century trends in Iowa watersheds: an investigation of observed and modelled hydrologic storages and fluxes in heavily managed landscapes. *International Journal of Climatology* 32, 9 (2012), 1373–1391. https://doi.org/10.1002/joc.2361

[26] William Stowe. 2016. Swimming Upstream: Des Moines Water Works Asks for Agricultural Accountability in A State that Claims to Feed the World. *ROOTSTALK — A Prairie Journal of Culture, Science and the Arts* 2, 1 (2016), 32–37. https://ojs.grinnell.edu/index.php/prairiejournal/article/view/312

[27] Virtual Institute-High Productivity Supercomputing. *VI-HPS: Score-p.* http://www.vi-hps.org/projects/score-p accessed on March 13, 2017.

[28] Tracy E. Twine, Jarod J. Bryant, Katherine T. Richter, Carl J. Bernacchi, Kelly D. McConnaughay, Sherri J. Morris, and Andrew D.B. Leakey. 2013. Impacts of elevated CO2 concentration on the productivity and surface energy budget of the soybean and maize agroecosystem in the Midwest USA. *Global change biology* 19, 9 (2013), 2838–2852.

[29] Peter C. Van Metre, Jeffrey W. Frey, MaryLynn Musgrove, Naomi Nakagaki, Sharon Qi, Barbara J. Mahler, Michael E. Wieczorek, and Daniel T. Button. 2016. High Nitrate Concentrations in Some Midwest United States Streams in 2013 after the 2012 Drought. *Journal of Environmental Quality* 45, 5 (2016), 1696–1704.

[30] Andy Vanloocke, Carl J. Bernacchi, and Tracy E. Twine. 2010. The impacts of Miscanthus× giganteus production on the Midwest US hydrologic cycle. *Gcb Bioenergy* 2, 4 (2010), 180–191.

[31] Andy VanLoocke, Tracy E. Twine, Christopher J. Kucharik, and Carl J. Bernacchi. 2016. Assessing the potential to decrease the Gulf of Mexico hypoxic zone with Midwest US perennial cellulosic feedstock production. *GCB Bioenergy* (2016). https://doi.org/10.1111/gcbb.12385

[32] Andy VanLoocke, Tracy E. Twine, Marcelo Zeri, and Carl J. Bernacchi. 2012. A regional comparison of water use efficiency for miscanthus, switchgrass and maize. *Agricultural and Forest Meteorology* 164 (2012), 82–95.

[33] Berkley J. Walker, Andy VanLoocke, Carl J. Bernacchi, and Donald R. Ort. 2016. The costs of photorespiration to food production now and in the future. *Annual review of plant biology* 67 (2016), 107–129.

[34] Donald J. Wuebbles and Katharine Hayhoe. 2004. Climate Change Projections for the United States Midwest. *Mitigation and Adaptation Strategies for Global Change* 9, 4 (2004), 335–363. https://doi.org/10.1023/B:MITI.0000038843.73424.de

[35] Qichun Yang, Hanqin Tian, Marjorie A.M. Friedrichs, Charles S. Hopkinson, Chaoqun Lu, and Raymond G. Najjar. 2015. Increased nitrogen export from eastern North America to the Atlantic Ocean due to climatic and anthropogenic changes during 1901–2008. *Journal of Geophysical Research: Biogeosciences* 120, 6 (2015), 1046–1068. https://doi.org/10.1002/2014JG002763 2014JG002763.