Scaling Parallel Modeling of Agroecosystems with Lustre

Tyler Balson[1], Yuwei Li[2], Adam Ward[1]
H.E. Cicada Dennis[1], Robert Henschel[1], Holger Brunst[3]
Stephen Simms[1], Shawn Slavin[1]

1 Indiana University
2 CuraCloud Corporation
3 ZIH, Technische Universität Dresden

Background

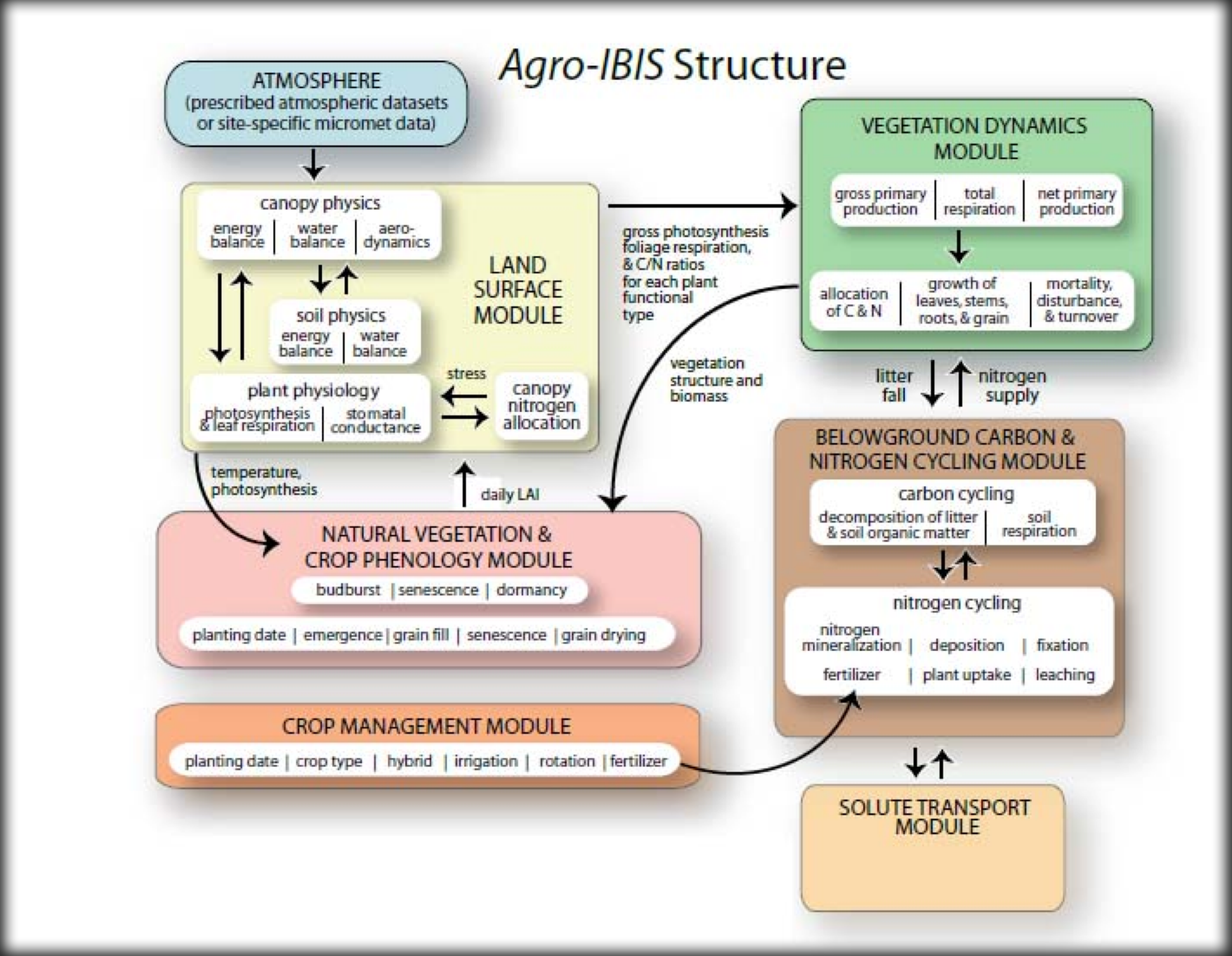
- Agricultural land management is a topic of ever-increasing complexity
  - The Midwest is an important source of international argiculture
  - Increasing demands on crop production put strain on resources
  - Crop production vs environmental impacts an area of increasing tension
  - Water use, land management, urban growth, climate change are all factors
- Tools to model, simulate, and predict interplay of geo, eco, and agro cycles provide important info for stakeholders

Background

- Agro-IBIS*
  - Simulates agricultural ecosystems
    - Inputs include climate and weather data, farming decisions, and landscape properties
    - Outputs include physical state variables, fluxes, and agricultural parameters
    - Widely validated results for Midwestern US
  - Serial, Fortran code written to run in a classic single-process mode
  - Data is available to simulate at much larger scale
  - Need to develop an HPC implementation of Agro-IBIS to solve large-scale models

* Kucharik, C.J., and K.R. Brye (2003), Journal of Environmental Quality, 32(1), 247-268; https://lter.limnology.wisc.edu/project/agro-ibis

Agro-IBIS Structure



ATMOSPHERE
(prescribed atmospheric datasets or site-specific micromet data)

LAND SURFACE MODULE

canopy physics
energy balance | water balance | aero-dynamics

soil physics
energy balance | water balance

plant physiology
photosynthesis & leaf respiration | stomatal conductance

stress

canopy nitrogen allocation

gross photosynthesis foliage respiration, & C/N ratios for each plant functional type

VEGETATION DYNAMICS MODULE

gross primary production | total respiration | net primary production

allocation of C & N | growth of leaves, stems, roots, & grain | mortality, disturbance, & turnover

vegetation structure and biomass

litter fall

nitrogen supply

temperature, photosynthesis

daily LAI

NATURAL VEGETATION & CROP PHENOLOGY MODULE

budburst | senescence | dormancy

planting date | emergence | grain fill | senescence | grain drying

BELOWGROUND CARBON & NITROGEN CYCLING MODULE

carbon cycling
decomposition of litter & soil organic matter | soil respiration

nitrogen cycling
nitrogen mineralization | deposition | fixation
fertilizer | plant uptake | leaching

CROP MANAGEMENT MODULE

planting date | crop type | hybrid | irrigation | rotation | fertilizer

SOLUTE TRANSPORT MODULE

Agro-IBIS at IU

- Agro-IBIS Development: Gains and Constraints
  - Strong desire to maintain consistency with community code
    - Optimizations desired to be drop-in or easily integrated with downloaded code
  - Implementation of netCDF library for standardized, optimized data storage
  - Parallel MPI wrapper written in C++ to manage domain decomposition and job launching
  - Previously unrecognized I/O bottleneck waiting to show up in parallel runs

Agro-IBIS Workflow

- Agro-IBIS was initially parallelized without regard to the impact on the file system
  - Each instance runs without communication to others
    - ”embarrasingly parallel” (aka “massively serial”)
    - File input and output parameters are managed by C++ MPI wrapper
    - IBIS instances are run via system() call from C++; literally no inter-process communication

Agro-IBIS Workflow

- Method for running IBIS puts a lot of strain on the file systems used
  - Inputs and outputs for each IBIS run are separate file trees
  - IBIS instances scale perfectly – if you could ignore I/O cost
  - Very easy to tax the MDS without realizing it
  - This is just an example of what any conventional, serial app would do when domain decomposition doesn't take I/O into account.

DC2 @ Indiana University

- Data Capacitor 2
  - 5.3 PB on Lustre 2.16 (2.7 upgrade in 06/17) on LDISKFS
  - 16 OSS/252 OSTs served by two DDN SFA12k; 1 active MDT
  - 56 Gbs FDR IB to HPC systems
  - Serves both project storage and scratch for IU’s HPC systems

Initial Challenges

- Scaling to a very large number of processes on the Cray (BigRed2 @ IU)
  - Straightforward to implement; lots of capacity on Cray
  - Initially run when our Lustre scratch fs (DC2) was redlining
  - Results
    - Scaling very disappointing for researchers
    - DC2 performance slowed to a crawl, disappointing everyone (esp admins!)
- First serious look at I/O in IBIS code revealed several inefficiencies

I/O Bottlenecks

- Maintenance day testing against DC2
  - Test cases were run from 02 – 16 nodes x 16 Agro-IBIS procs on IU’s Cray XE6
  - IOPS peaked at ~25000
  - Agro-IBIS processes write I/O peaked at 24 GB/s
  - Runs were made at 16 x 16 ppn for both 1x and 8x striping
    - Results were very similar with a slight improvement for single striping

DCRAM – Lustre with SSD

- Our short-term strategy was dictated by the scenario
- Parallel implementation of Agro-IBIS was dragging down DC2
- An experimental system with SSDs was available
- Originally envisioned to support biology apps with many small files
- Reconfigured to support Agro-IBIS use and relieve DC2 stress

DCRAM @ Indiana University

- DCRAM
  - 35 TB of SSD on Lustre 2.8.0 backed by LDISKFS
  - 6 OSS/12 OSTs
    - Each OST is a RAID-0 array of 4 x 800 GB Intel DC 3500 SSDs
  - 2 active MDS supporting DNE2
    - Each MDT (1 per MDS) is RAID-0 array of 4 800 GB Intel DC 3510 SSDs
    - We are currently limited to using DNE1 by Lustre 2.5.1 Cray clients
  - 40 Gbps FDR-10 IB to HPC systems

DCWAN @ Indiana University

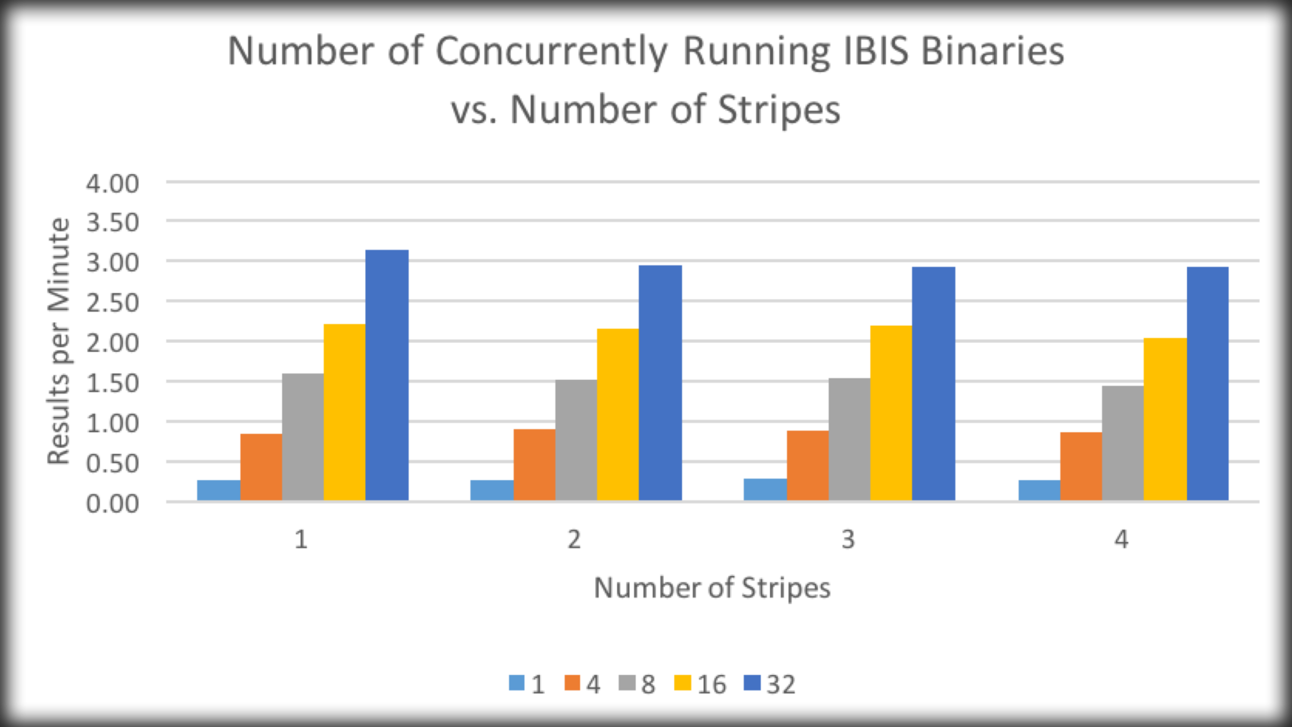
- DCWAN
  - 661 TB (currently) on Lustre 2.8.0 on ZFS
  - 4 OSS/109 OSTs served by DDN SFA10k; 1 active MDT on LDISKFS
  - 10/40 (MDS/OSS) Gbs Ethernet to campus network
  - Serves WAN-based projects and IU’s HPC systems, using nodemapping, uid/gid mapping, and Lustre SK on wire

Problem Analysis

- Vampir and Score-P were used to profile runtime
- We still do not have a good way to see into runtime Lustre activity
  - but we are working on this
  - DC2 is at Lustre 2.1.6, so no jobstats data available (yet)
  - DCRAM did not have jobstats enabled when testing was done, but does now
    - We still have not developed job_id-to-uid mapping for jobstats with Torque/PBS
  - Initial view showed what we expected
    - Huge number of file opens and closes, consistent with code written for hw 25 years ago
    - Lots and lots of small reads and writes

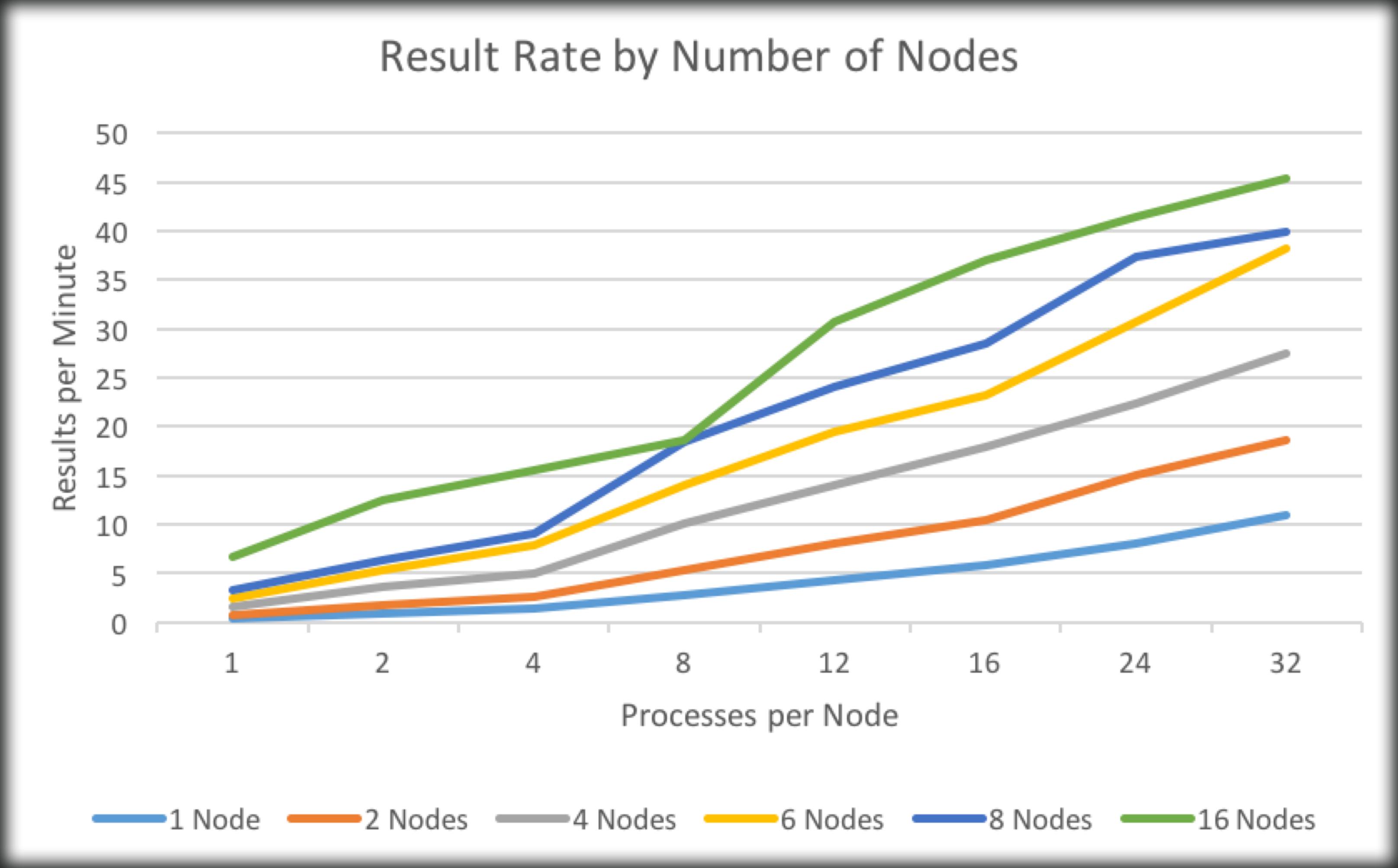
Solution Strategies

- To improve I/O perf we tried
  - Different stripe counts
    - Didn’t really help on reads
    - Actually hurt on writes
  - Changing stripe size
    - No real improvement

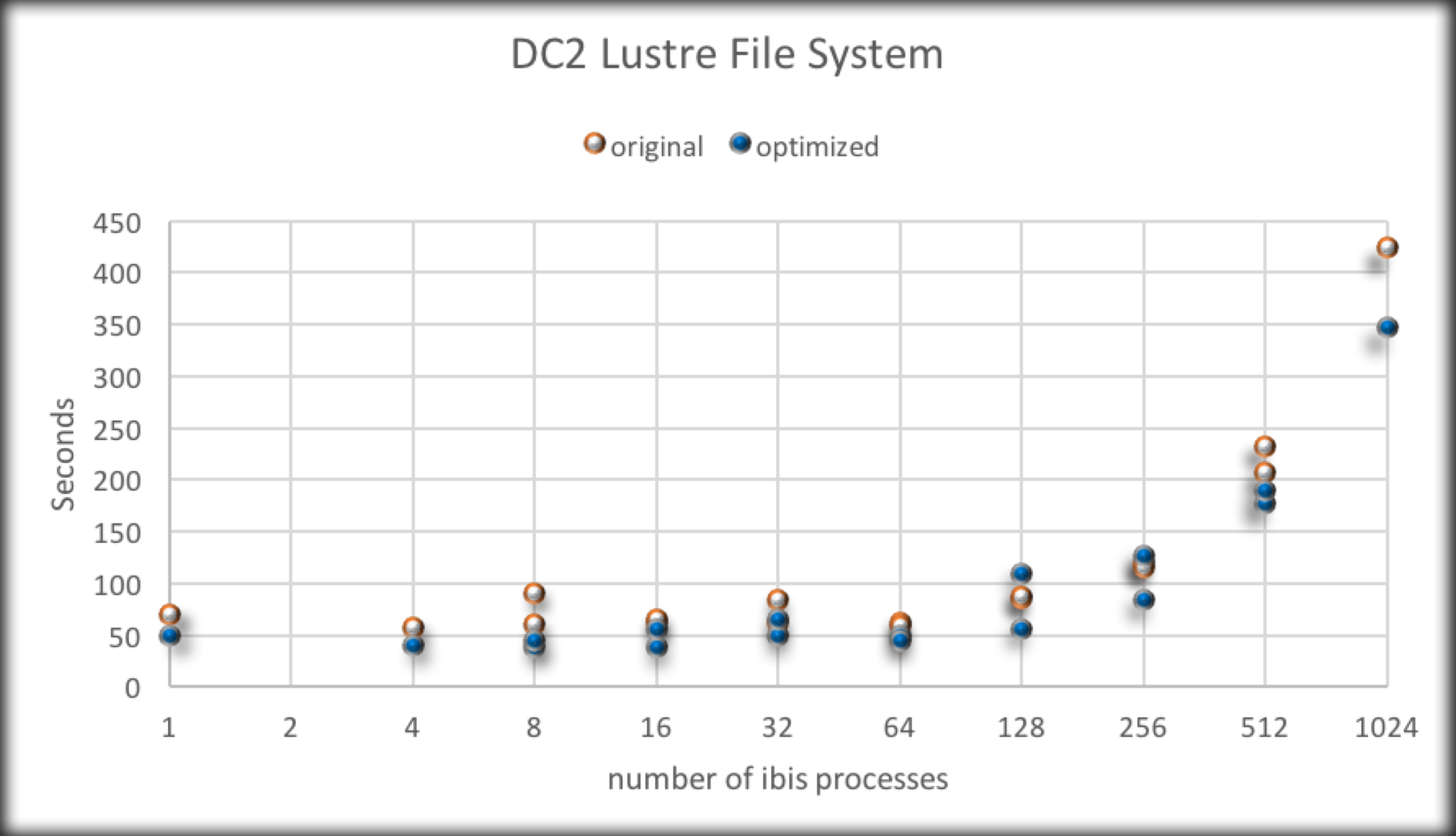


Solution Strategies

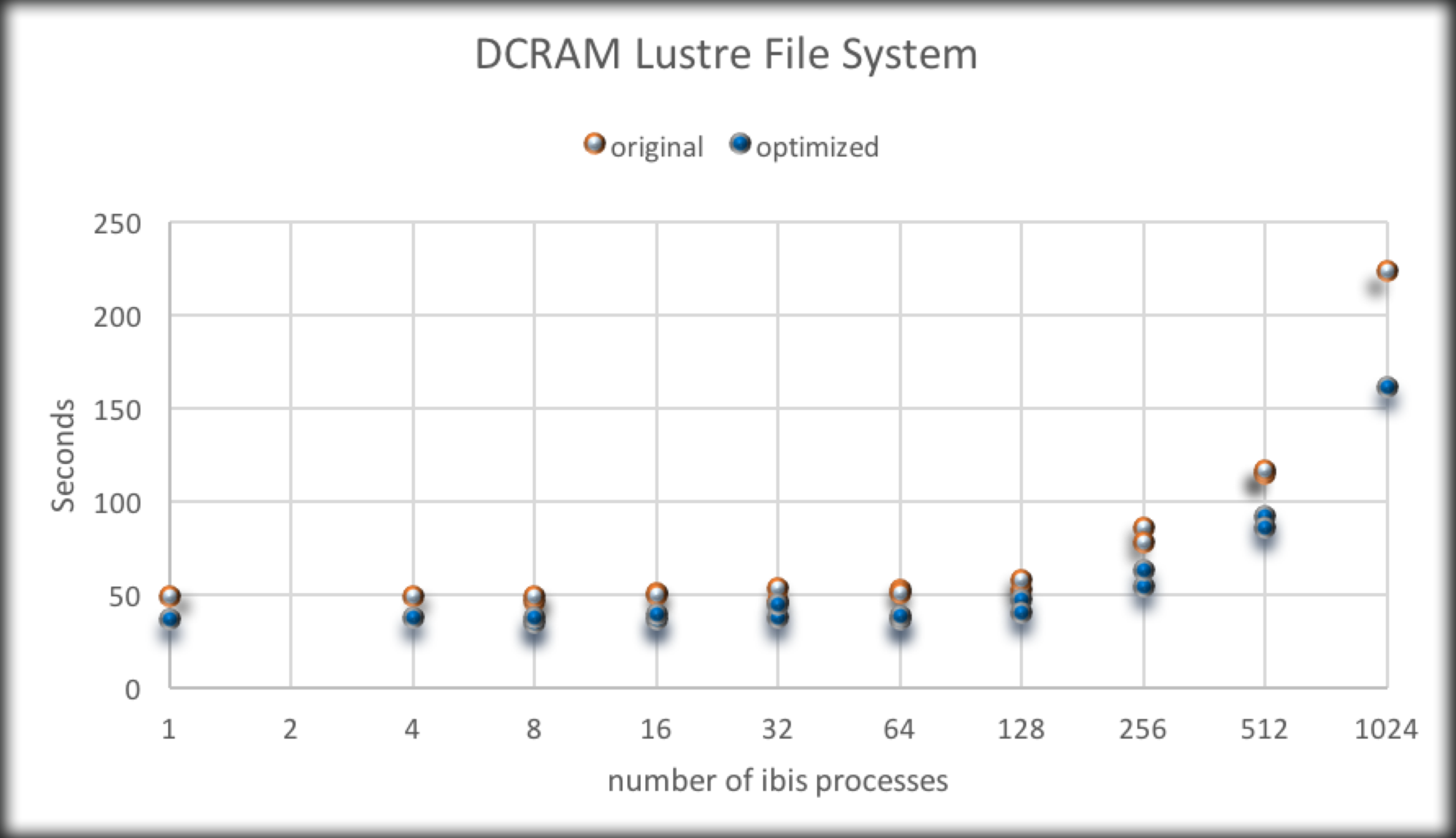
- To improve I/O performance we tried
  - netCDF “diskless” writes and large read file buffers win
    - Use of NC_DISKLESS|NC_WRITE with netCDF allows for in-memory files with write at close
    - 8 kB buffers (default) worked best for writes
    - 4 MB buffers for reads optimized read performance best
  - Code to minimize unnecessary opens and closes relieves strain on filesystem
  - Moving write activity to /tmp (SHM on BR2) & staging output to Lustre a big win



IBIS results per minute vs. PPN for runs with nodes = 1 – 16



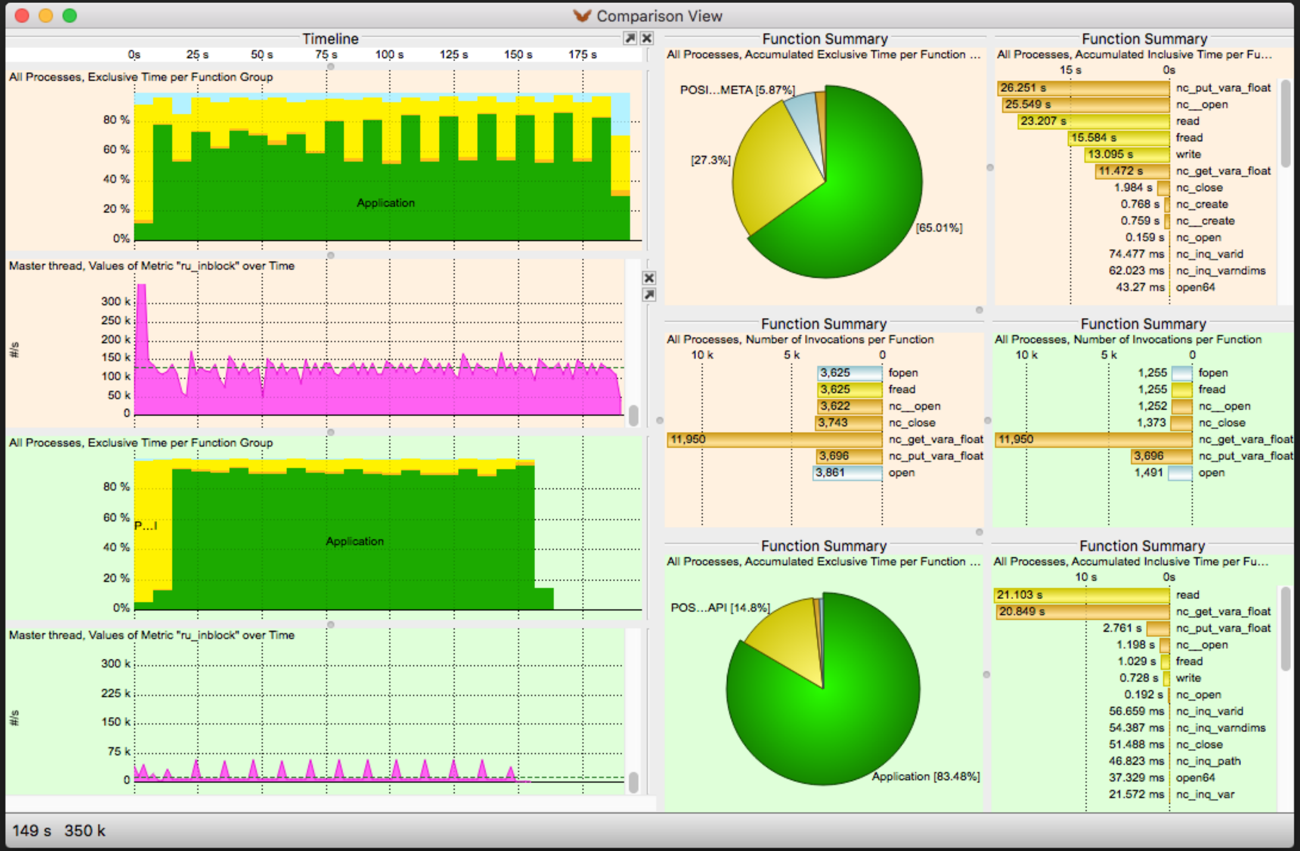
Agro-IBIS results on Big Red 2 with and without optimization

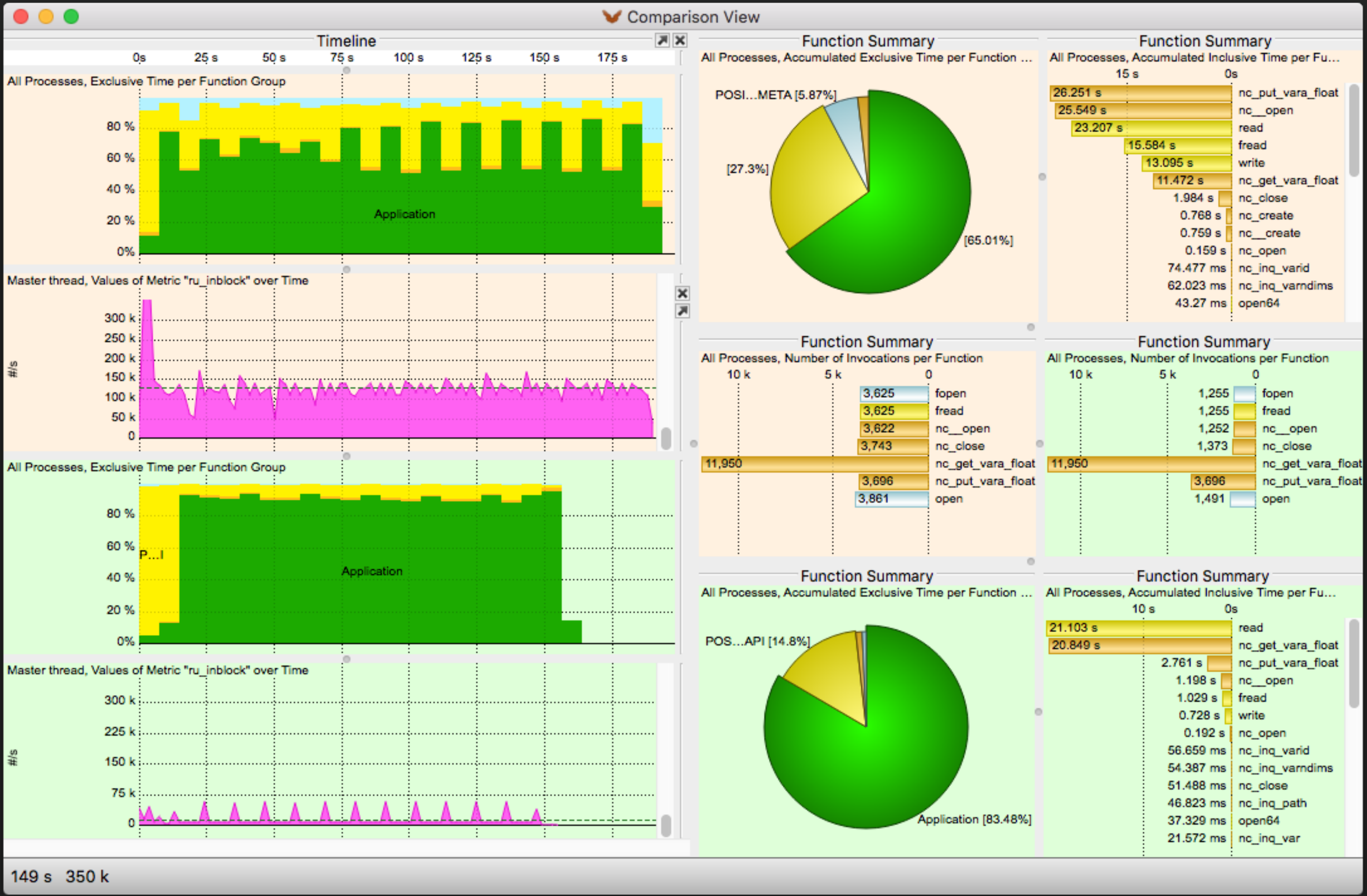


Agro-IBIS results on Big Red 2 with and without optimization

Profiling Agro-IBIS

- Another look at the optimization with Vampir provides insights
  - Orange background *not* optimized
  - Green background is optimized
- Optimization for minimal opens and closes clearly visible




Comparison View
Timeline
0s
25 s
50 s
75 s
100 s
125 s
150 s
175 s
All Processes, Exclusive Time per Function Group
80 %
60 %
40 %
20 %
0%
Application
Master thread, Values of Metric "ru_inblock" over Time
#/s
300 k
250 k
200 k
150 k
100 k
50 k
0
All Processes, Exclusive Time per Function Group
80 %
60 %
40 %
20 %
0%
P...I
Application
Master thread, Values of Metric "ru_inblock" over Time
#/s
300 k
225 k
150 k
75 k
0
Function Summary
All Processes, Accumulated Exclusive Time per Function ...
POSI...META [5.87%]
[27.3%]
[65.01%]
Function Summary
All Processes, Accumulated Exclusive Time per Function ...
POS...API [14.8%]
Application [83.48%]
149 s 350 k

Function Summary

All Processes, Accumulated Inclusive Time per Fu...

15 s 0s	
26.251 s	nc_put_vara_float
25.549 s	nc__open
23.207 s	read
15.584 s	fread
13.095 s	write
11.472 s	nc_get_vara_float
1.984 s	nc_close
0.768 s	nc_create
0.759 s	nc__create
0.159 s	nc_open
74.477 ms	nc_inq_varid
62.023 ms	nc_inq_varndims
43.27 ms	open64

Function Summary

All Processes, Number of Invocations per Function

10 k 5 k 0	
3,625	fopen
3,625	fread
3,622	nc__open
3,743	nc_close
11,950	nc_get_vara_float
3,696	nc_put_vara_float
3,861	open

Function Summary

All Processes, Number of Invocations per Function

10 k 5 k 0	
1,255	fopen
1,255	fread
1,252	nc__open
1,373	nc_close
11,950	nc_get_vara_float
3,696	nc_put_vara_float
1,491	open

Function Summary

All Processes, Accumulated Inclusive Time per Fu...

10 s 0s	
21.103 s	read
20.849 s	nc_get_vara_float
2.761 s	nc_put_vara_float
1.198 s	nc__open
1.029 s	fread
0.728 s	write
0.192 s	nc_open
56.659 ms	nc_inq_varid
54.387 ms	nc_inq_varndims
51.488 ms	nc_close
46.823 ms	nc_inq_path
37.329 ms	open64
21.572 ms	nc_inq_var

Key Points

1. Use of SSD-based Lustre filesystem a huge help to our researchers
2. It remains true that maximizing local runtime I/O is best strategy
   - Not a huge help that our Cray has only ramdisk for /tmp (limited to 32 GB)
3. Lustre caching and prefetching seems to work well enough that our optimizations on buffering have very limited effect
4. Attempts to put old serial codes into an HPC context must consider I/O
   - This project has required a group of people with widely differing skill sets

Future Work

1. Further development of parallel Agro-IBIS implementation – in progress
    - Will require reprogramming of parts of the IBIS implementation
    - Implementing IBIS as an MPI application with some I/O managed as messages
    - Use of MPI-IO extensions in netCDF in post-processing of data zones
2. Planning movement of application back to DC2 and successors
    - DCRAM is not large enough, nor redundant

Thank You!

Questions?