IN53A-1721 A New Parallel Library and Scripting Language for Analysis of Ultra-large Climate Data Sets

parVis

Robert Jacob (jacob@mcs.anl.gov), Jayesh Krishna, Xiabing Xu, Sheri Mickelson, Tim Tautges, Mike Wilde, Rob Latham, Ian Foster, Rob Ross, Mark Hereld, (Argonne National Laboratory); Pavel Bochev, Kara Peterson, Mark Taylor (Sandia National Lab.); Karen Schuchardt, Jian Yin (Pacific Northwest National Lab.); Don Middleton, Mary Haley, David Brown, Richard Brownrigg, Wei Huang, Dennis Shea, Mariana Vertenstein (NCAR), Kwan-Liu Ma, Jinrong Xie (UC-Davis)

Problem:

Climate model output continues to grow in size from both increased resolution and additional fields

- CAM HOMME at 0.125 degrees
  - Single 3D variable: 616 MB
  - Single 2D variable: 25 MB
  - Single history file: 24 GB
  - 1 year of monthly output: 288 GB
  - 100 years of monthly: 28.8 TB
- CSU GCRM 4km horizontal, 100 levels
  - Single 3D variable (cell center): 16 GB
  - Single 3D variable (cell edge): 50.3 GB
  - Single history file 571 GB
  - 1 year of monthly output: 6 TB
  - 100 years of monthly: .6 PB

Climate models are going beyond their traditional structured, rectangular grids. Ex: atmosphere cubed sphere grid (left), Spherical Centroidal Voronoi Tessellation (right)

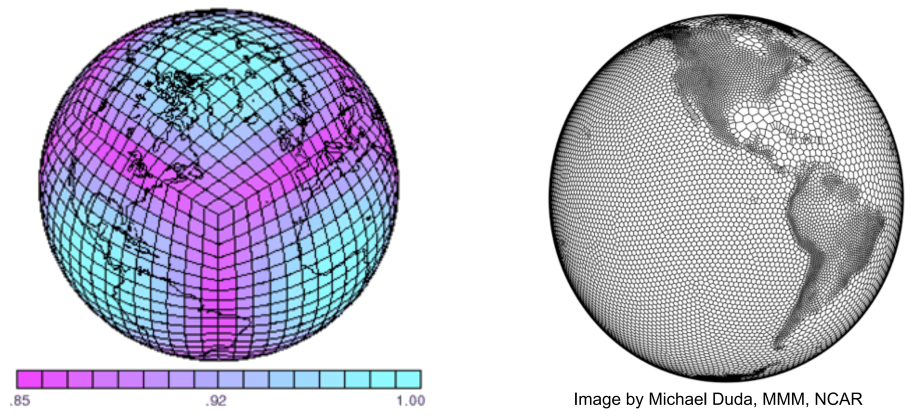
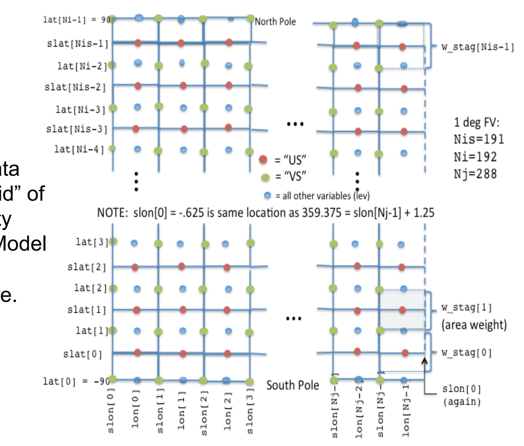


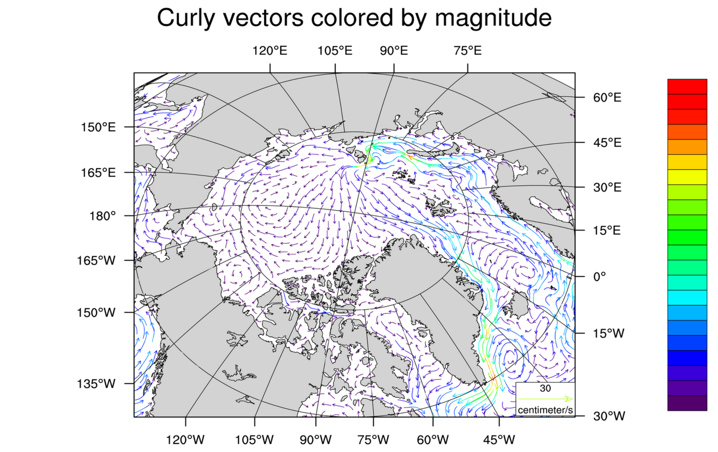
Image by Michael Duda, MMM, NCAR

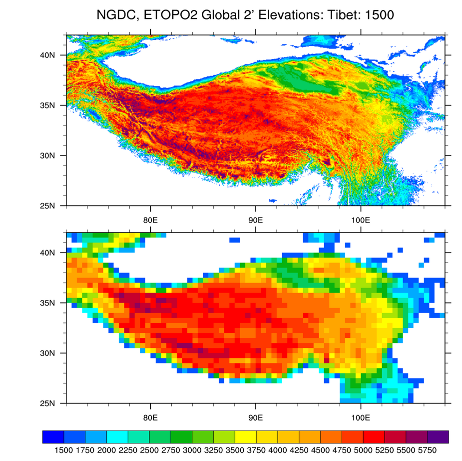
Analysis packages are often not aware of where data lies on grid.

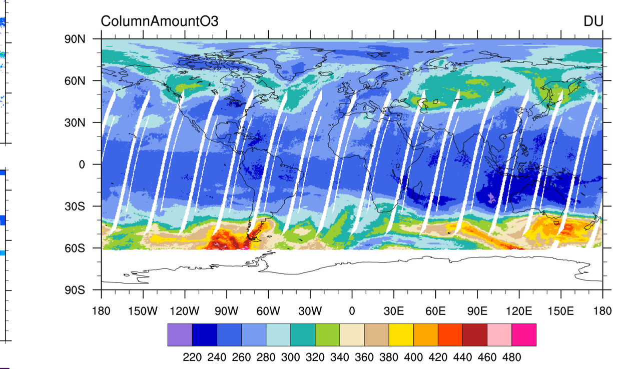


Location of data on the "CD-grid" of the Community Atmosphere Model Finite Volume dynamical core.

Climate has a very low aspect ratio (10,000km wide while only 10km deep) This means 2D plots are the best way to view climate output. HOWEVER, the current tools to analyze and plot the data are single-threaded and assume rectangular grids!

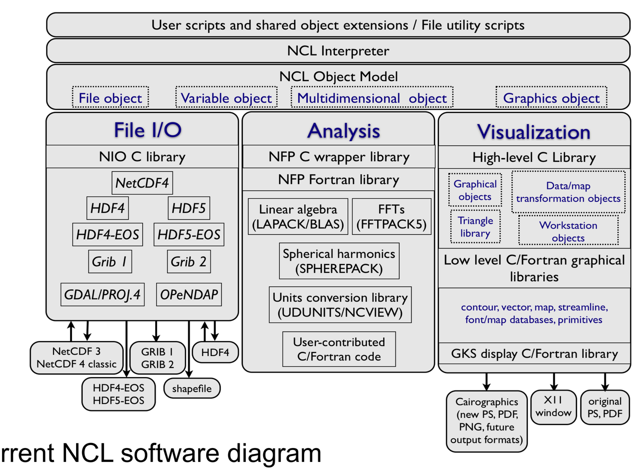






Elements of our solution

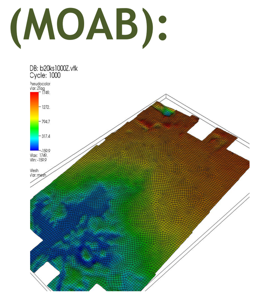
NCL (NCAR Command Language) is a widely used scripting language tailored for the analysis and visualization of geoscientfic data.



Current NCL software diagram

Mesh-Oriented datABase (MOAB):

- MOAB is a library for representing structured, unstructured, and polyhedral meshes, and field data on those meshes.
- Uses array-based storage, for memory efficiency
- Supports MPI-based parallel model



Glacier ice bed with VisIt/MOAB

INteroperable Tools for Rapid dEveloPment of compatIble Discretizations

Intrepid:

- An extensible library for computing operators on discretized fields
- Cell topology, geometry and integration
- Discrete spaces, operators and functionals on cell worksets
- Up to order 10 *H(grad)*, *H(curl)* and *H(div)* FE bases on Quad, Triangle, Tetrahedron, Hexahedron, and Wedge cell topologies
- High quality cubature, e.g., positive weights only on Tri and Tet cells

PNetCDF: NetCDF output with MPI-IO

Based on NetCDF
- Derived from their source code
- API slightly modified
- Final output is indistinguishable from serial NetCDF file

Additional Features
- Noncontiguous I/O in memory using MPI datatypes
- Noncontiguous I/O in file using sub-arrays
- Collective I/O

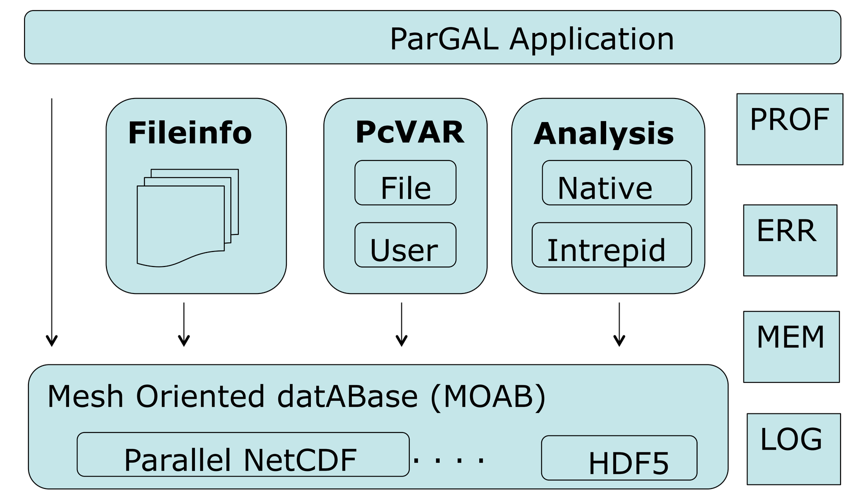
swift

- **Swift is a parallel scripting system for Grids and clusters**
- **Swift is easy to write**: simple high-level C-like functional language
- **Swift is easy to run**: a Java application. Just need a Java interpreter installed.
- **Swift is fast**: Karajan provides Swift a powerful, efficient, scalable and flexible execution engine.
- Swift is supported by an NSF SI2 grant.

ParNCL beta has been released! Download from `http://trac.mcs.anl.gov/projects/parvis`

ParGAL: Parallel Gridded Analysis Library

Using MOAB, Intrepid and PnetCDF, we have created the Parallel Gridded Analysis Library. ParGALprovides data-parallel versions of many typical analysis functions on gridded data sets. ParGAL functions can also operate on the native grids of the output and on the discretizations of the original model.

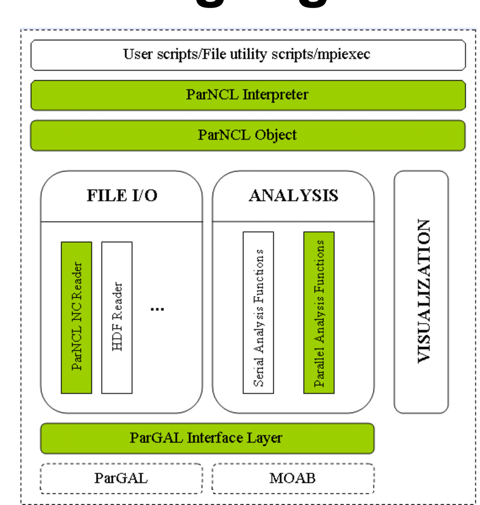


ParGAL Function table (so far)

NCL function Group	NCL Functions	ParGAL Function
File IO	addfile, addfiles	fileinfo, pcvar
Spherical Harmonic Routines	dv2uv* (4 funcs)	divergence
Meteorology	uv2dv_cfd	divergence
Spherical Harmonic Routines	uv2dv* (4 funcs)	divergence
Meteorology	uv2vr_cfd	vorticity
Spherical Harmonic Routines	uv2vr* (4 funcs)	vorticity
Spherical Harmonic Routines	uv2vrdv* (4 funcs)	vorticity, divergence
General Applied Math	dim_avg, dim_avg_n	dim_avg_n
General Applied Math	dim_max, dim_max_n	dim_max_n
General Applied Math	dim_min, dim_min_n	dim_min_n
General Applied Math	max	max
General Applied Math	min	min
Variable Manipulators	delete	pcvar
		gather

ParNCL: Parallel version of the NCAR Command Language

ParNCL uses ParGAL, MOAB and NCL graphics to create a parallel version of NCL. ParNCL is a parallel program that executes normal NCL scripts using parallel methods and data types. ParNCL allows operations on data sizes larger then the memory of a single node.



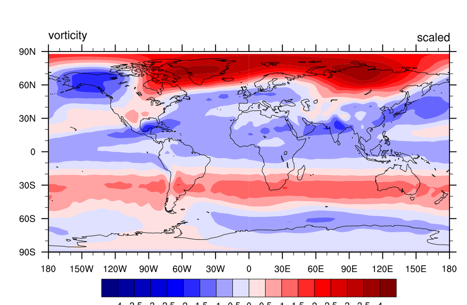
An NCL script executed by ParNCL

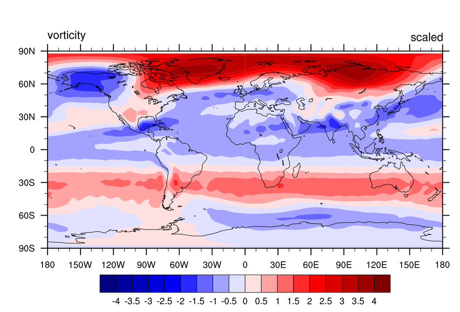
```
f = addfiles(diri+fili, "r”) ;   open file
tt = f[:]->T(0,{500},{-30:30},:) ;  read a section of data.

wks  = gsn_open_wks("ps","parvis_t")  ; open a PS file

plot = gsn_csm_contour_map(wks, tt(:,:),False)
```

Read with parallel I/O

a parallel data object

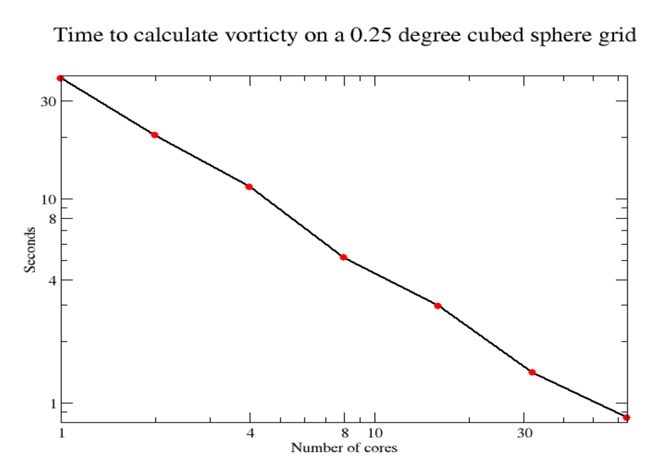
In the ParNCL interpreter, tt is gathered to one node and passed to normal NCL graphics routines for plotting.





- ParNCL (w/ ParGAL and Intrepid) version of diagnosed vorticity
- Calculated locally on each element
- Easily parallelizable
- Global data not required

- Original NCL version
- Uses spherical harmonics
- Requires global data



Using ParNCL requires a parallel environment:

Now: `> ncl myscript.ncl`

With ParNCL: `> mpirun -np 32 parcnl myscript.ncl`

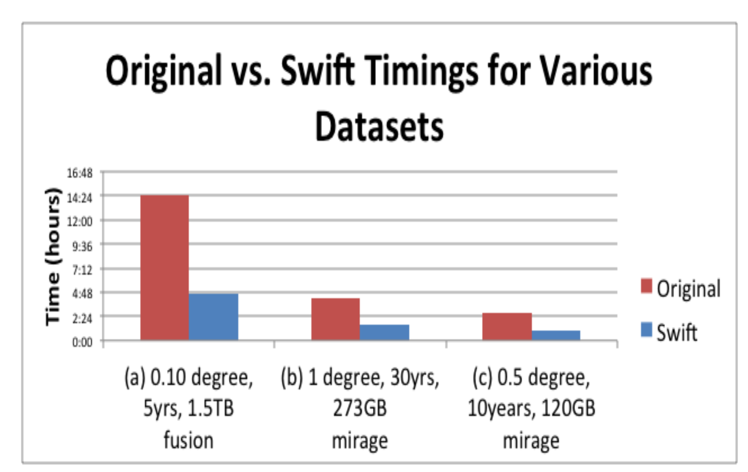
Total execution time for calculating vorticity directly on a 0.25 degree cubed sphere grid from the CAM-SE model.

Also, Task-parallelism with Swift

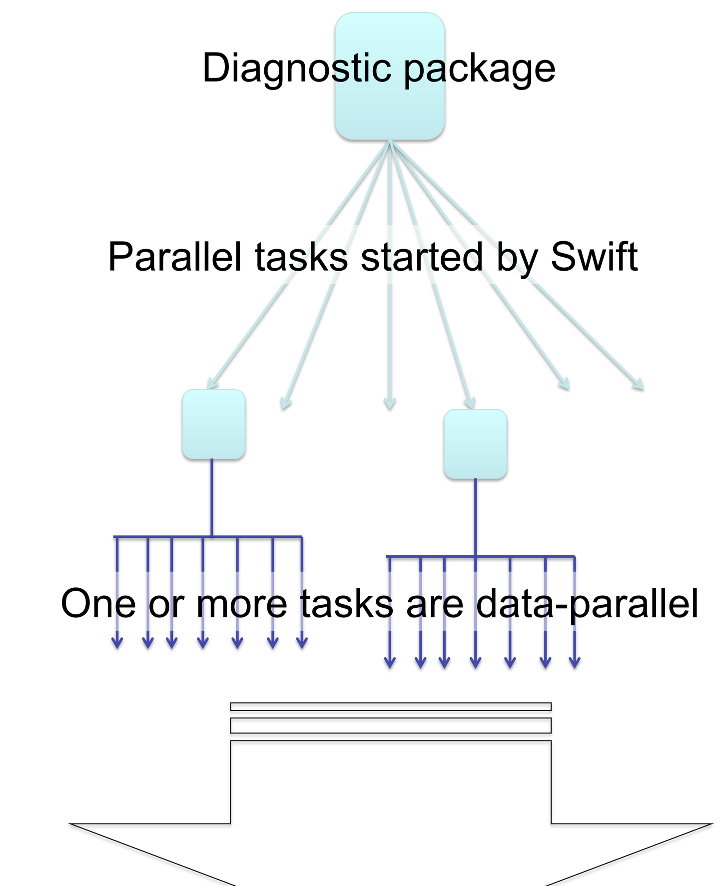
The various working groups of the DOE/NSF Community Earth System Model have each defined a suite of diagnostics that can be applied to the output of each climate model simulation. Hundred of plots are produced by each package. After modifying the scripts with Swift designate which parts can be done separately, the Swift engine can execute the tasks in parallel.

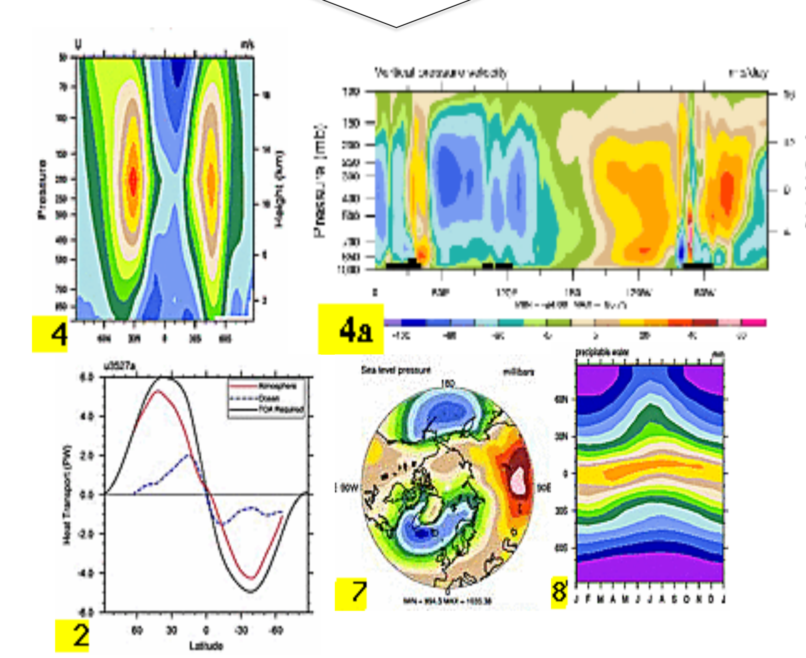
Swift-ized diagnostics with various data sets:

a) 5 years of 0.1 degree atm data
b) 30 years of 1 degree data
c) 10 years of 0.5 degree monthly data.

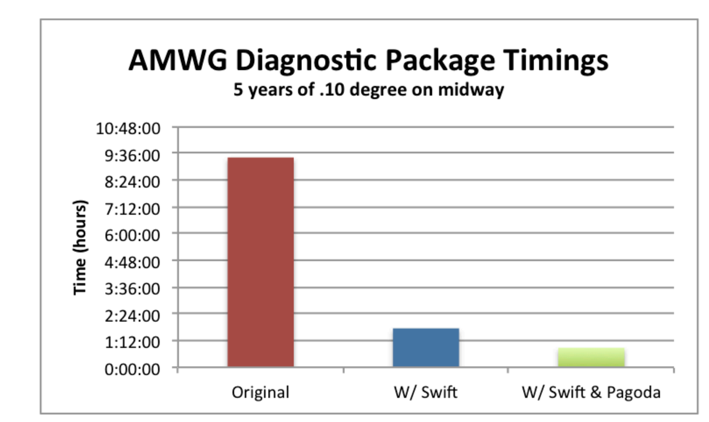


Future: Combining Data and Task-Parallelism





Using Pagoda, which provides data-parallel versions of NCO tools, we can combine task and data parallelism to complete diagnostics even faster.



Timings performed on "midway" of the University of Chicago's Research Computing Center.