ParallelIO Update

Jim Edwards,
NCAR CGD Software Engineer

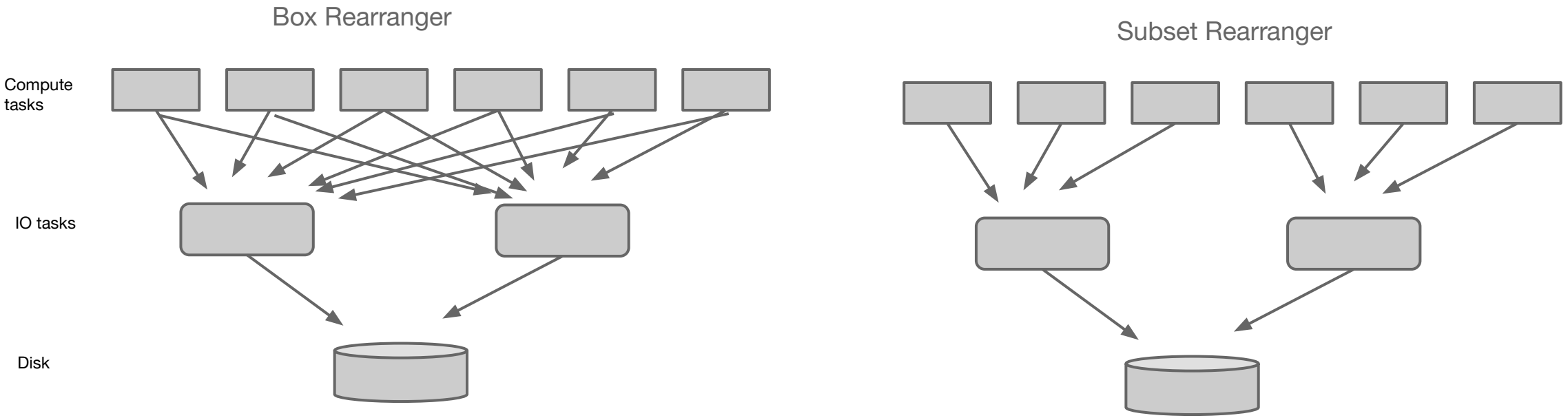
June 14, 2023

This material is based upon work supported by the National Center for Atmospheric Research, which is a major facility sponsored by the National Science Foundation under Cooperative Agreement No. 1852977.

What is ParallelIO?

ParallelIO is an interface library between CESM (and other Geophysical Models) and NetCDF and/or PnetCDF IO libraries.

It rearranges data read from files to the compute tasks that need it and from compute tasks back to files.



ParallelIO: What's new?

- Current release: 2.6.0
- NetCDF integration
- HDF5 Filter support
- Asynchronous capable

ParallelIO: NetCDF integration

- Available with version 4.9.2 or newer of netcdf-c
- Allows developers to convert easily from NetCDF API to ParallelIO API
- Able to use all available output formats

```
call MPI_Init(ierr)
call MPI_Comm_rank(MPI_COMM_WORLD, my_rank, ierr)
call MPI_Comm_size(MPI_COMM_WORLD, ntasks, ierr)

! Define an IOSystem.
ierr = nf_def_iosystem(my_rank, MPI_COMM_WORLD, niotasks,  &
        numAggregator, stride, PIO_rearr_box, iosysid, base)

ierr = nf_def_decomp(iosysid, PIO_int, dims, compdof, decompid)

! Create a file.
ierr = nf_create(FILE_NAME, NF_PIO, ncid)

! Define dimensions.
ierr = nf_def_dim(ncid, LAT_NAME, NLAT, var_dim(1))
ierr = nf_def_dim(ncid, LON_NAME, NLON, var_dim(2))
ierr = nf_def_dim(ncid, REC_NAME, NF_UNLIMITED, var_dim(3))

! Define a data variable.
ierr = nf_def_var(ncid, VAR_NAME, NF_INT, NDIM3, var_dim, varid)
ierr = nf_enddef(ncid)
  ! Write 1st record with distributed arrays.
ierr = nf_put_vard_int(ncid, varid, decompid, 1, data_buffer)
```

```
! Close the file.
ierr = nf_close(ncid)

! Free resources.
ierr = nf_free_decomp(decompid)
ierr = nf_free_iosystem()
deallocate(compdof)
deallocate(data_buffer)

! We're done!
call MPI_Finalize(ierr)
```

ParallelIO: HDF5 Filter and data compression support

- Only available with NetCDF4/HDF5 format
- Allows data to pass through user defined filters on the way to or from disk
  - filters can be pipelined so output of one filter becomes the input of the next filter
- Several predefined filters are available (depending on support in hdf5 and netcdf libraries)
  - zstandard
  - quantize (lossy compression: bitgroom, bitround, granularbr)
  - bzip2
  - deflate

- pio_inq_filter_avail
- pio_def_var_filter
- pio_def_var_deflate
- pio_def_var_quantize
- pio_def_var_bzip2
- pio_def_var_zstandard
- pio_def_var_szip

PIO: Asynchronous IO

- Functionally implemented
  - Tested in CESM2.3
    - ERS_Ld5.ne30pg3_t061.B1850MOM.cheyenne_intel.allactive-defaultio--drv-asyncio1node
    - ERS_Ln9.f19_f19_mg17.FHIST.cheyenne_intel.drv-asyncio1pernode--cam-outfrq9s
    - ERS_C3_Vnuopc.f19_g17.A.cheyenne_intel.drv-asyncio1pernode

Additional effort required to make the implementation performant.

One per node example:

PIO_ASYNCIO_NTASKS: 1
PIO_ASYNCIO_ROOTPE: 1
PIO_ASYNCIO_STRIDE: 128
PIO_ASYNC_INTERFACE: ['CPL:TRUE', 'ATM:TRUE', 'LND:TRUE', 'ICE:TRUE', 'OCN:TRUE', 'ROF:TRUE', 'GLC:TRUE', 'WAV:TRUE', 'ESP:TRUE']

One node example:

PIO_ASYNCIO_NTASKS: 4
PIO_ASYNCIO_ROOTPE: 0
PIO_ASYNCIO_STRIDE: 1
PIO_ASYNC_INTERFACE: ['CPL:TRUE', 'ATM:TRUE', 'LND:TRUE', 'ICE:TRUE', 'OCN:FALSE', 'ROF:TRUE', 'GLC:TRUE', 'WAV:TRUE', 'ESP:TRUE']

ParallelIO: future plans

Data compression can be specified with no changes in the component library.

Implement a configuration input file:

- Regex match on filename.
- Regex match on variable name.
- Allow user to specify data compression of variable.

ParallelIO: future plans

GPU Direct IO:

Model runs on compute tasks - asynchronous IO is used with ParallelIO on GPU's
Data writes use GPU for data compression operations
Resulting data is written to disk using GPU Direct IO

Proof of concept testing done in March 2023 NOAA/NCAR open hackathon.
Special thanks to Ben Kirk (CISL) and Brian Dobbins.

GPU Direct IO feature will be available on Derecho after legacy GPFS is retired (Dec 2023)
Available now for experiments on Casper.

ParallelIO: future plans

GPU Direct IO:

Known outstanding issues:

- Data compression methods implemented on GPUs must be compatible with those on CPUs.
- HDF5 Filter methods need work to keep data on GPUs

ParallelIO: Where to get it

Github: https://github.com/NCAR/ParallelIO/

ParallelIO is also available in:

- Spack
- Conda-forge
- EZ-Build

ESMF Aware Threading

Default threading model for CESM requires that all components running sequentially (using the same mpi tasks) must be threaded in the same way.

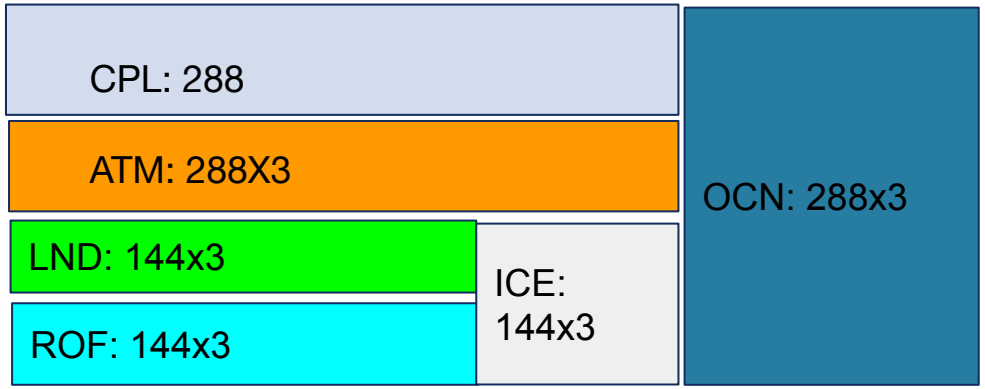
Using ESMF_AWARE_THREADING=TRUE provides more flexibility by allowing each component to be individually customized for optimal performance.

Changes the definition of ROOTPE from MPI task of driver to MPI task of ESMF.

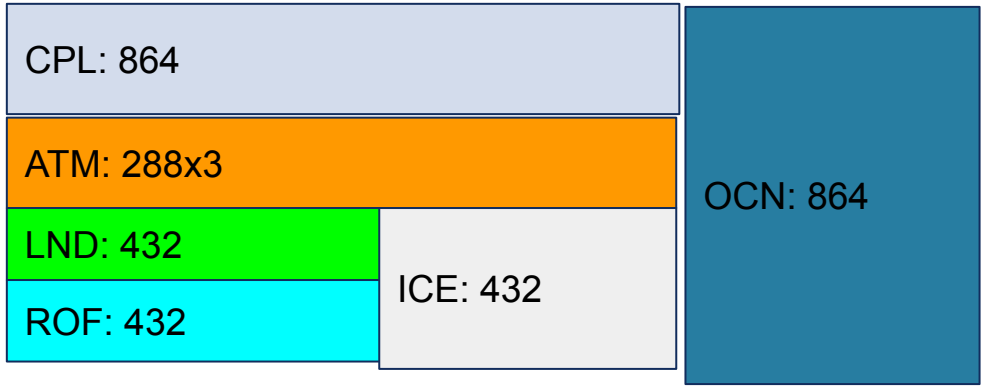
```
./pelayout
Comp  NTASKS  NTHRDS  ROOTPE PSTRIDE
CPL :   4096/       1;        0              1
ATM :   1024/       4;        0              1
LND :   1024/       1;        0              1
ICE :   3072/       1;     1024              1
OCN :   256/        1;     4096              1
ROF :   256/        1;        0              4
GLC :   128/        4;        0              1
WAV :   128/        4;        0              1
ESP :     1/        4;        0              1
ESMF_AWARE_THREADING is True
ROOTPE is with respect to 128 tasks per node
```

ESMF Aware Threading

default threading



ESMF aware threading



Questions?