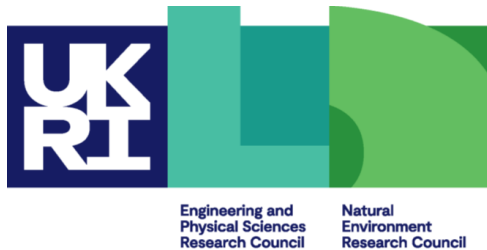# Advanced Message-Passing Programming

Alternative Parallel IO Libraries

Reusing this material

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

http://creativecommons.org/licenses/by-nc-sa/4.0/

This means you are free to copy and redistribute the material and adapt and build on the material under the following terms: You must give appropriate credit, provide a link to the license and indicate if changes were made. If you adapt or build on the material you must distribute your work under the same license as the original.

Note that this presentation contains images owned by others. Please seek their permission before reusing these images.

Overview

- Issues with MPI-IO

- HDF5

- NetCDF

- ADIOS2

- Summary

MPI-IO Issues

- Files are raw bytes
  - no header information
  - storage is architecture-specific (e.g. big / little-endian floating-point)

- Difficult to deal with in other codes downstream
  - user must write their own post-processing tools

- But ...
  - it can be very fast!

Solution

- For functionality
  - define higher-level formats
  - include metadata, e.g. "this is a 4x5x7 array of doubles"
  - enables standard data converters, browsers, viewers etc.

- For performance
  - layer on top of MPI-IO

- Many real applications use higher-level formats
  - understanding MPI-IO will enable you to get performance as well

HDF5

- "**Hierarchical Data Format** (**HDF**) is a set of file formats (**HDF4**, **HDF5**) designed to store and organize large amounts of data." (Wikipedia)
  - data arranged like a Unix file system
  - self-describing
  - hierarchical
  - can use MPI-IO

image taken from www.hdfgroup.org
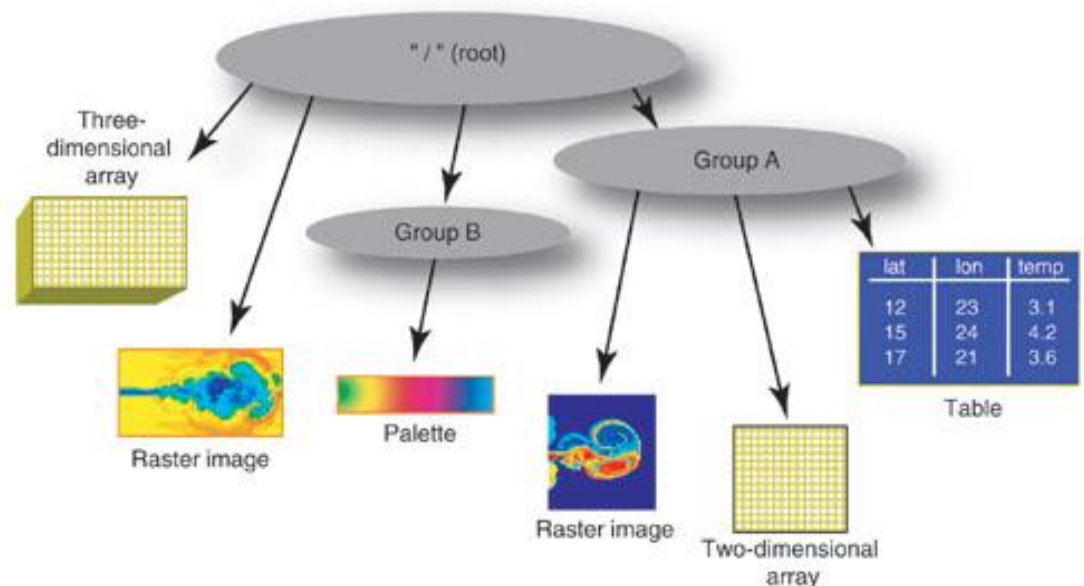
Parallel HDF5 (Fortran)

- Approach much like MPI-IO
  - describe global dataset
  - each process describes its local portion(s) of the global data, encodes sizes

MPI_ORDER_ FORTRAN

global data, encodes sizes

```
CALL h5sselect_hyperslab_f(filespace, &
        H5S_SELECT_SET_F, offset, &
        count, error)
```

starts

- Then call collective write
  - hyperslabs can be merged to create global file
  - actual file IO done through MPI-IO
  - important to choose collective IO

subsizes

NetCDF: Network Common Data Form

- "a set of [software libraries](#) and self-describing, machine-independent data formats that support the creation, access, and sharing of [array-oriented](#) scientific data.." (Wikipedia)
  - more restricted than HDF5
  - common in certain communities
    - climate research
    - oceanography
    - GIS ...

- Rich set of tools
  - data manipulation
  - visualisation
  - ...

txxETCCDI_yr_MIROC5_historical_r2i1p1_1850-2012.nc

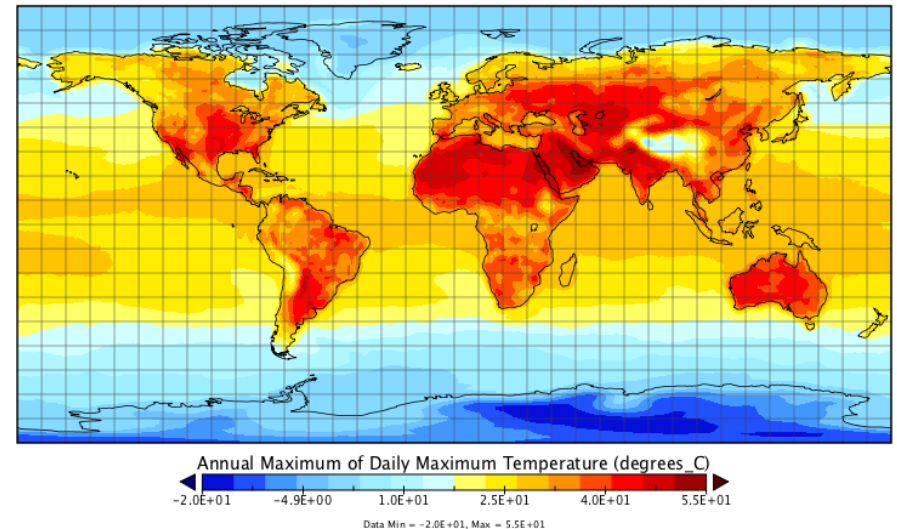Annual Maximum of Daily Maximum Temperature

Annual Maximum of Daily Maximum Temperature (degrees_C)

-2.0E+01   -4.9E+00   1.0E+01   2.5E+01   4.0E+01   5.5E+01

Data Min = -2.0E+01, Max = 5.5E+01

image taken from http://live.osgeo.org

Parallel NetCDF (Fortran)

file identifier

sizes

```fortran
nf90_def_var(ncid, "data", NF90_DOUBLE, dimids,
varid) )

...

nf90_var_par_access(ncid, varid, nf90_collective)

...

nf90_put_var(ncid, varid, buf, start, count)
```

Write_all()

starts

subsizes

ADIOS2

- A recent IO parallel library https://adios2.readthedocs.io/
  - can output using native MPI-IO or HDF5
  - also supports its own formats, e.g. BP5 (binary-pack v5)
- Same overall approach
  - each process defines what portion(s) of global data it owns
  - call read/write routines
- Much more configurable at runtime via "config.xml"
  - e.g. no need to recompile to switch MPI-IO to BP5
- Now part of benchio and colleagues have seen potential benefits from BP5 format in other codes
  - writes multiple files under the hood (with associated metadata)
  - may get round the limits of MPI-IO which has a single shared file

ADIOS2 (Fortran)

```fortran
! Define the global array
  call adios2_define_variable(var_g, io, "GlobalArray", adios2_type_dp, &
                              ndim, arraygsize, arraystart, arraysubsize, &
                              adios2_constant_dims, ierr)


! Begin ouput step
  call adios2_begin_step( bp_writer, ierr)

  call adios2_put( bp_writer, var_g, out_data, ierr)

! End the output
  call adios2_end_step(bp_writer, ierr)
```
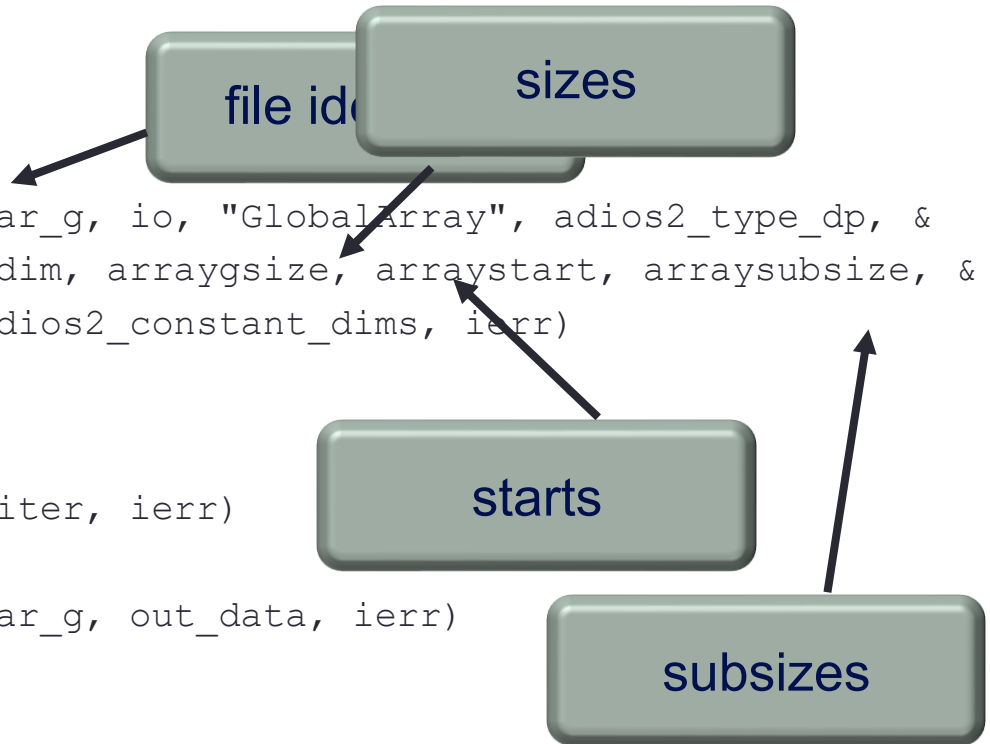
file id

sizes

starts

subsizes

How is ADIOS2 different?

- For MPI-IO, HDF5 and NetCDF
  - output file is independent of process count or decomposition
  - requires a lot of data movement
  - locking issues as multiple aggregators write to the same shared file
- For ADIOS2 "BP5" format
  - under the hood, writes one file per aggregator
    - no data movement
    - no locking issues
  - also stores metadata to make this invisible to the user
  - most file options are stored in an XML file
    - can change output format to HDF5 without recompilation

Summary

- MPI-IO may seem a little low-level
  - but is fundamental building block of parallel IO on most systems

- Higher-level formats layer on top of MPI-IO
  - to benefit from performance work by vendors, Lustre etc.

- Common formats are HDF5 and NetCDF
  - both supported on ARCHER2
  - you might also want to look at the newer ADIOS2 library

- Understanding MPI-IO performance is key to getting good performance for HDF5 and NetCDF