Space and Performance Optimizations of NetCDF-Grids

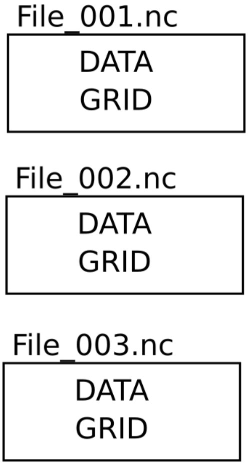
Eugen Betke, Julian Kunkel, Jakob Luettgau

Research Group
German Climate Computing Center

2017-09-26

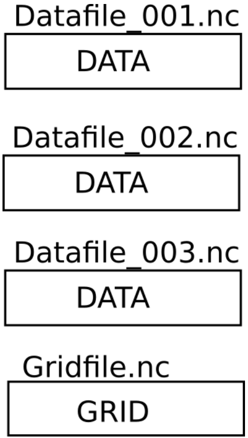
Table Of Content

Grid and Data in One File



- Advantages
  - Data and grid are connected
  - Assignment of dimensions to variables is not necessary
- All disadvantages of redundancy
  - Increased usage of storage space and network bandwidth
  - Inconsistencies are possible

Grid and Data in Separate Files



- Advantages
  - Guaranteed consistency
  - Optimal usage of storage space and network bandwidth
  - Grid can be moved to a fast storage tier
    - Resulting in short loading times
    - Makes sense, because a grid is often reused
- Disadvantages
  - Data and grid are not connected
  - Assignemt of dimensions to variables is necessary

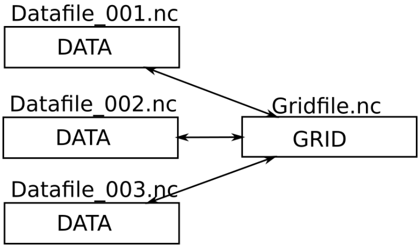
Examples from $HD(CP)2$ Model

- Model: $HD(CP)2$ [1]
- Grid and data are in separate files

Resolution in [cells]	Grid size in [GB]	Data size in [GB]
2323968	2.8	100
7616120	9.1	340
22282304	27	1000

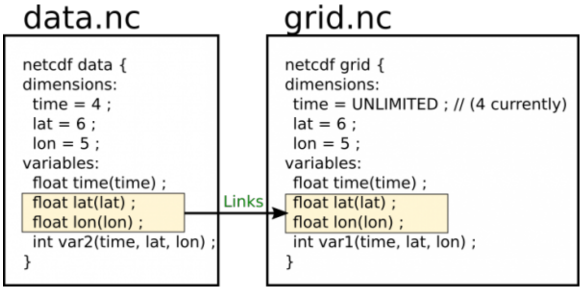
[1] High definition clouds and precipitation for advancing climate prediction

Linked Grid and Data



- Grid and data are
  - stored in separate files
  - linked together
- Links are transparent to application
- Data files are backwards compatible

NetCDF-Patch



- Requires HDF5 Virtual Datasets (> HDF5-1.10.0)
- Extends NetCDF4 interface with external dimensions
- Details you find on our project webpage:
  - `https://wr.informatik.uni-hamburg.de/research/projects/bullio/netcdf_external_links/start`

Table Of Content

Motivation: GRID-I/O on Exascale HPC

- Assumption: Exascale HPC will be build of
  - Heterogenous storage tiers, e.g.
    - Burst Buffers, Lustre, . . .
    - NVRAM, SSDs, HDDs, . . .
  - Large amount of cores
    - $>$ 1.000.000
- Consequence: Large number of small I/O accesses
  - One of the most suboptimal I/O patterns for many storage technologies

Overview: Adaptive Tiering Approach

- Intelligent middleware
- Usage of several different storage technologies in an efficient way
- Redirection of I/O accesses to best fitted tiers, e.g:
  - small I/O accesses to a fast, expensive storage tier (e.g. Burst Buffer)
  - large I/O accesses to a slow, cheap storage tier (e.g. HDD-based Lustre)

Grid Example from *HD*(*CP*)2 Model

```
$ h5ls GRID_3d_fine_DOM03_ML_20130502T000000Z.nc
bnds                         Dataset {2}
clat                         Dataset {22282304}
clat_vertices                Dataset {22282304, 3}
clon                         Dataset {22282304}
clon_vertices                Dataset {22282304, 3}
height                       Dataset {151}
height_2                     Dataset {150}
height_2_bnds                Dataset {150, 2}
height_bnds                  Dataset {151, 2}
ncells                       Dataset {22282304}
topography_c                 Dataset {22282304}
vertices                     Dataset {3}
z_ifc                        Dataset {151, 22282304}
z_mc                         Dataset {150, 22282304}
```

- Variables have different sizes
- Models surface of the earth
- NetCDF4 file

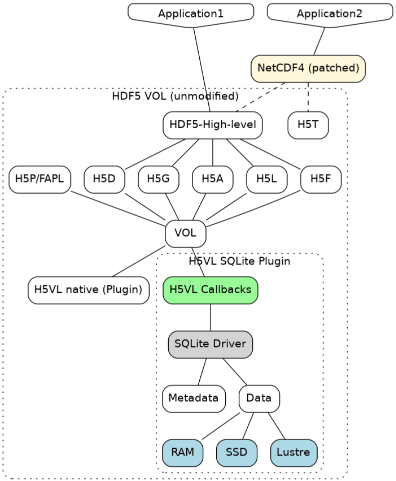
Data Example from *HD*(*CP*)2 Model

```
bnds                    Dataset {2}
ccb                     Dataset {360/Inf, 22282304}
cct                     Dataset {360/Inf, 22282304}
clch                    Dataset {360/Inf, 1, 22282304}
clcl                    Dataset {360/Inf, 1, 22282304}
clcm                    Dataset {360/Inf, 1, 22282304}
clivi                   Dataset {360/Inf, 22282304}
clt                     Dataset {360/Inf, 22282304}
clwvi                   Dataset {360/Inf, 22282304}
graupel_gsp_rate        Dataset {360/Inf, 22282304}
hail_gsp_rate           Dataset {360/Inf, 22282304}
hbas_con                Dataset {360/Inf, 22282304}
htop_con                Dataset {360/Inf, 22282304}
htop_dc                 Dataset {360/Inf, 22282304}
ice_gsp_rate            Dataset {360/Inf, 22282304}
lev                     Dataset {1}
lev_2                   Dataset {1}
lev_2_bnds              Dataset {1, 2}
lev_3                   Dataset {1}
lev_3_bnds              Dataset {1, 2}
lev_bnds                Dataset {1, 2}
ncells                  Dataset {22282304}
prw                     Dataset {360/Inf, 22282304}
rain_gsp_rate           Dataset {360/Inf, 22282304}
snow_gsp_rate           Dataset {360/Inf, 22282304}
t_cbase                 Dataset {360/Inf, 22282304}
t_ctop                  Dataset {360/Inf, 22282304}
time                    Dataset {360/Inf}
vertices                Dataset {3}
z_pbl                   Dataset {360/Inf, 22282304}
```

- A data file contains a set of variables
- Variables have different sizes
- Most dataset values are associated with a position in the grid
- In the application, variables and dimensions must be attached to each other, because grid is located in an external file

Adaptive Tiering Prototype

- Based on HDF5-VOL implementation
  - svn co `https://svn.hdfgroup.uiuc.edu/hdf5/features/vol/`
- Published under LGPL3 on GitHub
  - `https://github.com/ESiWACE/esdm`
- Separates Metadata and Data
  - Metadata on Lustre
  - Data on RAM, SSD, Lustre
- Writes/Reads data to/from best fitted storage tier



NetCDF Performance Benchmark Tool

- Developed at DKRZ
- Written in C
- Published on GitHub under LGPL License
  - `https://github.com/joobog/netcdf-bench`
- Mimics scientific data
  - Data is written in time steps

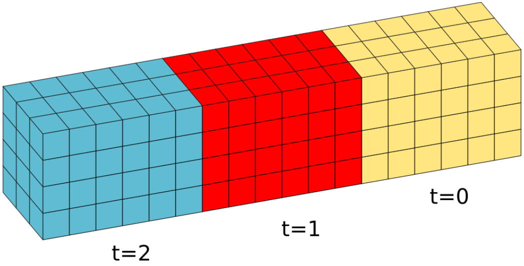


Figure: Data geometry (3:6:4:3)

NetCDF-Bench Example Output

```
$ mpiexec -n 1 ./benchtool
Benchtool (datatype: int)
Data geometry (t:x:y:z x sizeof(type))      100:100:100:10 x 4 bytes
Block geometry (t:x:y:z x sizeof(type))       1:100:100:10 x 4 bytes
Datasize                                          40000000 bytes                (40.0 MB)
Blocksize                                           400000 bytes                (400.0 kB)
I/O Access                                     independent
Storage                                         contiguous
File length                                          fixed
File value                                              no
                                                                      min                avg                max
benchmark:write      Open time                                0.2811507931       0.2811507931       0.2811507931 secs
benchmark:write      I/O time                                 0.1901479111       0.1901479111       0.1901479111 secs
benchmark:write      Close time                               0.3576489800       0.3576489800       0.3576489800 secs
benchmark:write      I/O Performance (w/o open/close)       200.6173638152     200.6173638152     200.6173638152 MiB/s
benchmark:write      I/O Performance                         46.0185526612      46.0185526612      46.0185526612 MiB/s
```

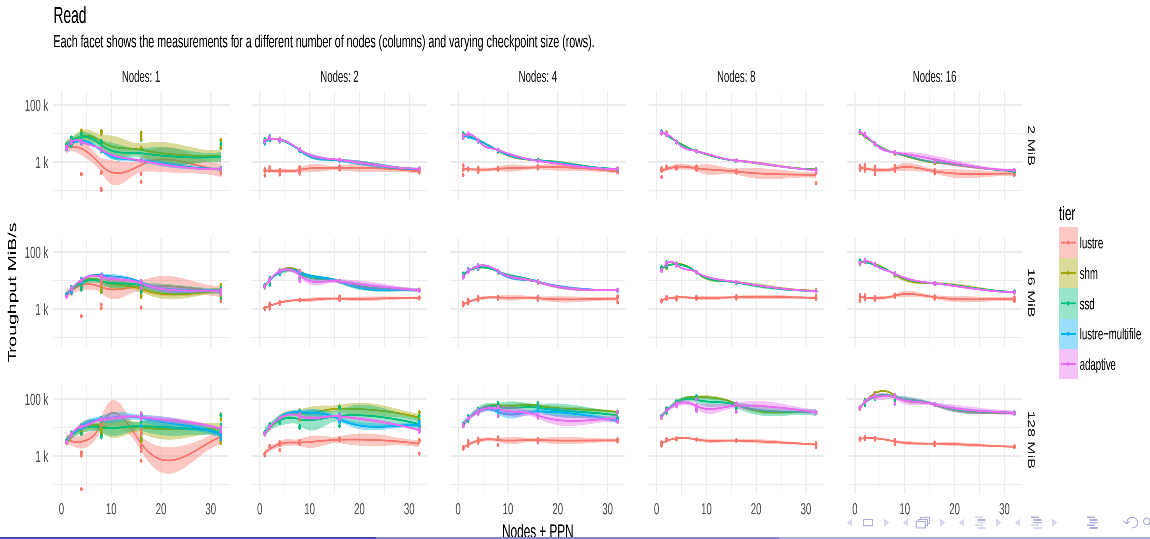
Features

- Semi-automatic domain decomposition
- Independent/Chunked/Collective I/O
- Pre-filling with fill value
- Limited/Unlimited dimensions support
- Aggregates results
  - Measures Open/IO/Close times
  - Computes I/O performance
  - Provides min/avg/max values for each measurement

Experiment Setup

- Experiments were conducten on Mistral (HPC of DKRZ)
- Storage technologies used in the experimentes are RAM, SSD, HDD-based Lustre
  - RAM peak performance about 4GB/s
  - SSD peak performance about 500 MB/s
  - Lustre peak performance is about 450 GbB/s
- Nodes: 1, 2, 4, 8, 16
- Processes: 1, 2, 4, 8, 16, 36
- Datasize in [MiB]: 2, 16, 128

Experiment with Adaptive Tiering Read



Experiment with Adaptive Tiering Write

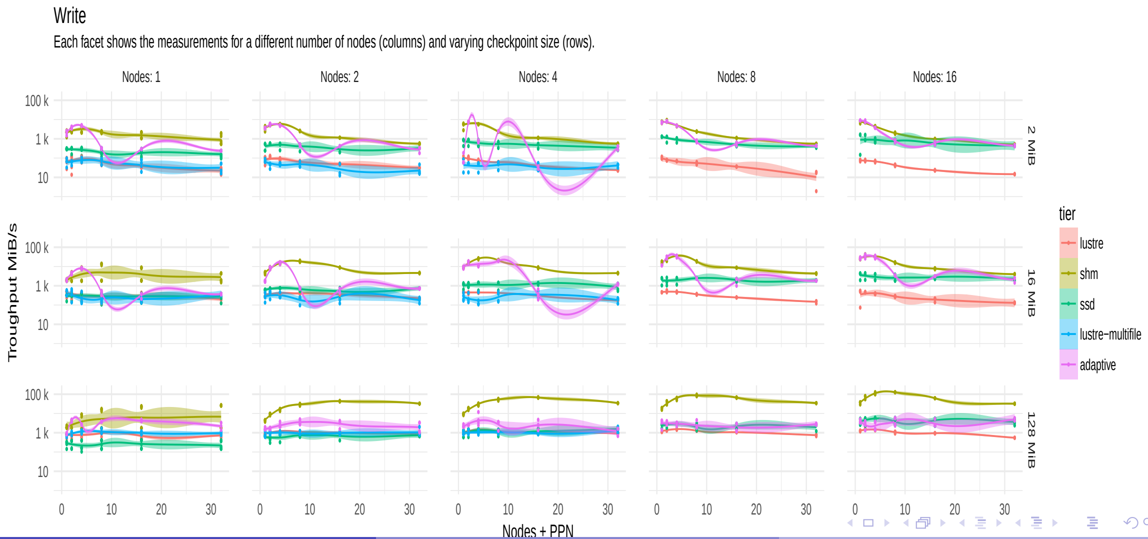


Table Of Content

Discussion

- Local vs. shared Grids
  - Save storage space
  - Can be moved to fast storage tiers for fast loading
- Intelligent middleware
  - Redirects automatically object I/O (e.g. access to variables) to optimal storage tiers
    - Transparent to the user/application

Discussion

- Shared Grids
  - Do we really need grid sharing in NetCDF?
  - How many people need such a feature NetCDF?
  - Shall it a default feature in NetCDF?
- I/O performance of unstructured grids on exascale HPC
  - What do you think, is loading/storing grids a problem, in respect to I/O performance, or is it exaggerated in this presentation?