Multi Dataset I/O

September 30, 2022

Neil Fortner, The HDF Group

The HDF Group

Review: HDF5 File Structure

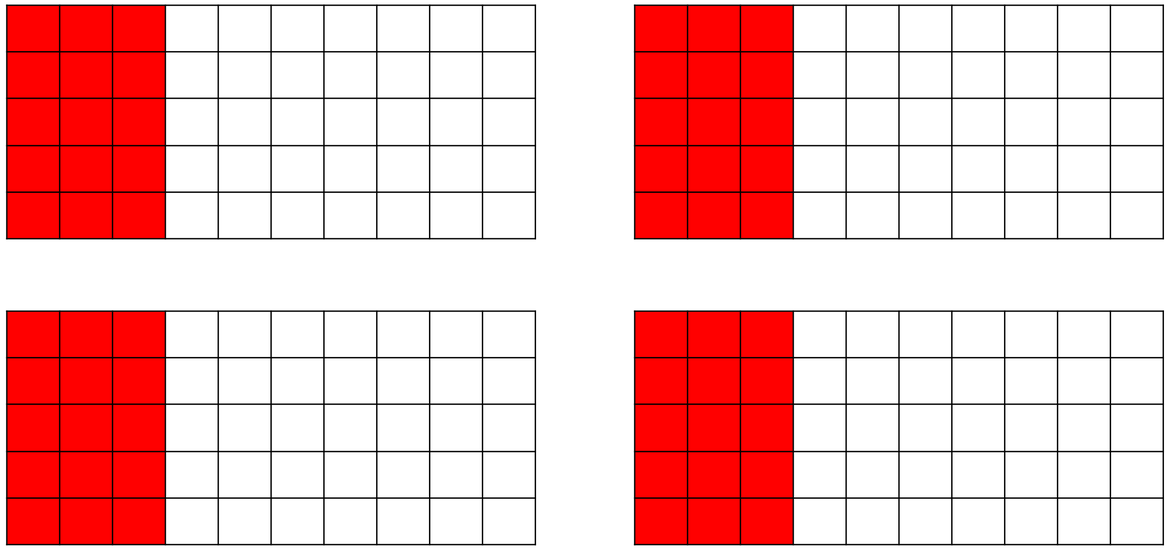
- Data in an HDF5 is stored file in **datasets**
- An HDF5 file can consist of any number of datasets, which can describe different variables, conditions, etc.
- I/O on datasets is currently performed via H5Dread() and H5Dwrite(), and is limited to one dataset at a time
- Datasets can be broken up into multiple **chunks**, which are stored separately in the file
- I/O can be performed on a subset of the dataset and a subset of the memory buffer, using dataspace selections

Current Status: Parallel Raw Data I/O

- **Independent**
  - Each rank performs I/O independently, no coordination between processes
  - No time lost due to waiting for synchronization
  - Cannot call MPI_File_set_view() since that is a collective operation, therefore must call MPI_File_read/write_at() for each contiguous block in the I/O
  - Potential performance loss because we aren't giving MPI the full picture of the I/O that is to be performed – both due to the need for contiguous blocks and because we aren't giving MPI I/O info from all ranks

Independent I/O Example

- **Chunked dataset with partial I/O (red squares):**
  - One MPI_File_read/write_at() call per row, so **20** calls total

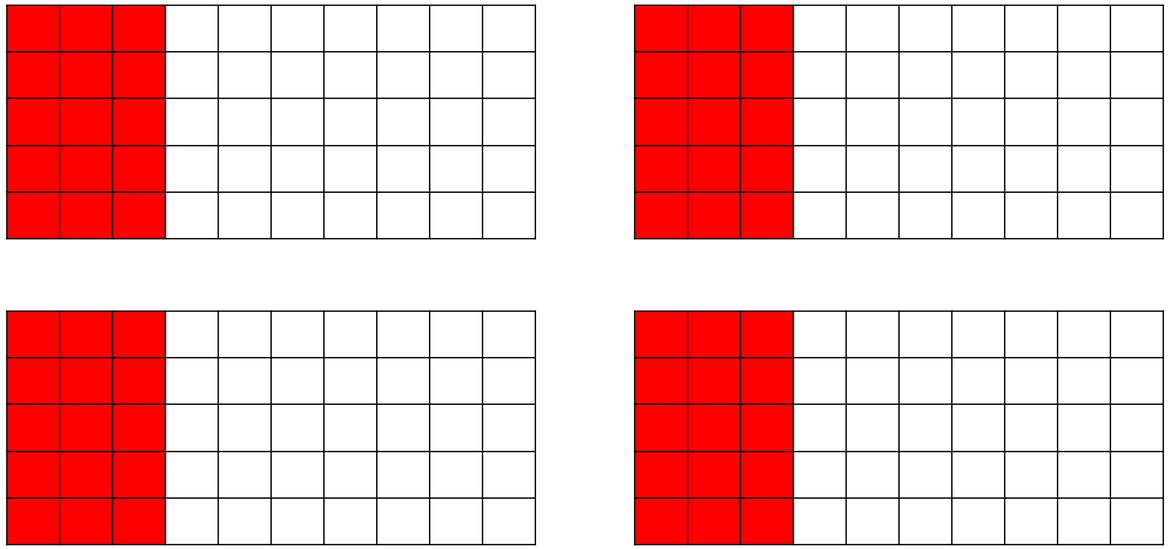


Current Status: Parallel Raw Data I/O

- **Multi-Chunk Collective**
  - HDF5 iterates over each chunk in the I/O, with each rank creating its own file view for each, and each rank issuing a single MPI_File_read/write_at(_all)() call for each chunk
  - Underlying I/O can be independent or collective (chosen automatically by default)
  - Gives more information to MPI than independent I/O pathway, while avoiding complex MPI datatypes

Multi-Chunk I/O Example

- **Chunked dataset with partial I/O (red squares):**
  - One MPI_File_read/write_at(_all)() call per chunk, so **4** calls total

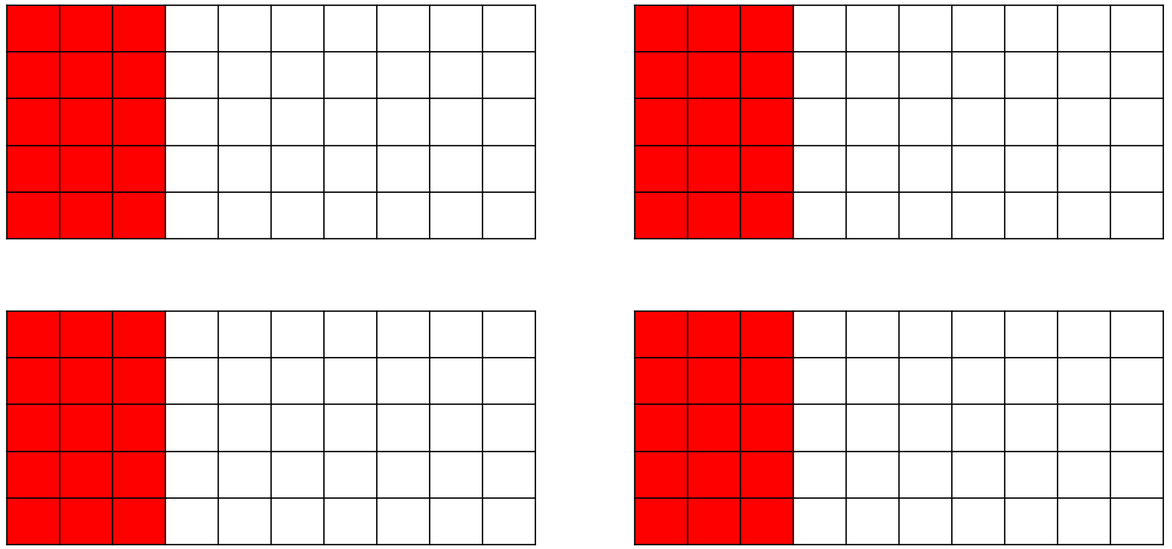


Current Status: Parallel Raw Data I/O

- **Link-Chunk Collective**
  - HDF5 builds complicated MPI datatypes that describe the entire I/O, spanning all chunks involved
  - Underlying I/O can be independent or collective (chosen automatically by default)
  - Single call to MPI_File_read/write_at(_all)() per I/O (one dataset per I/O)
  - Gives the maximum amount of information to MPI possible given the current H5Dread/write() APIs

Link-Chunk I/O Example

- **Chunked dataset with partial I/O (red squares):**
  - One MPI_File_read/write_at(_all)() call per I/O, so **1** call total



Multi Dataset I/O

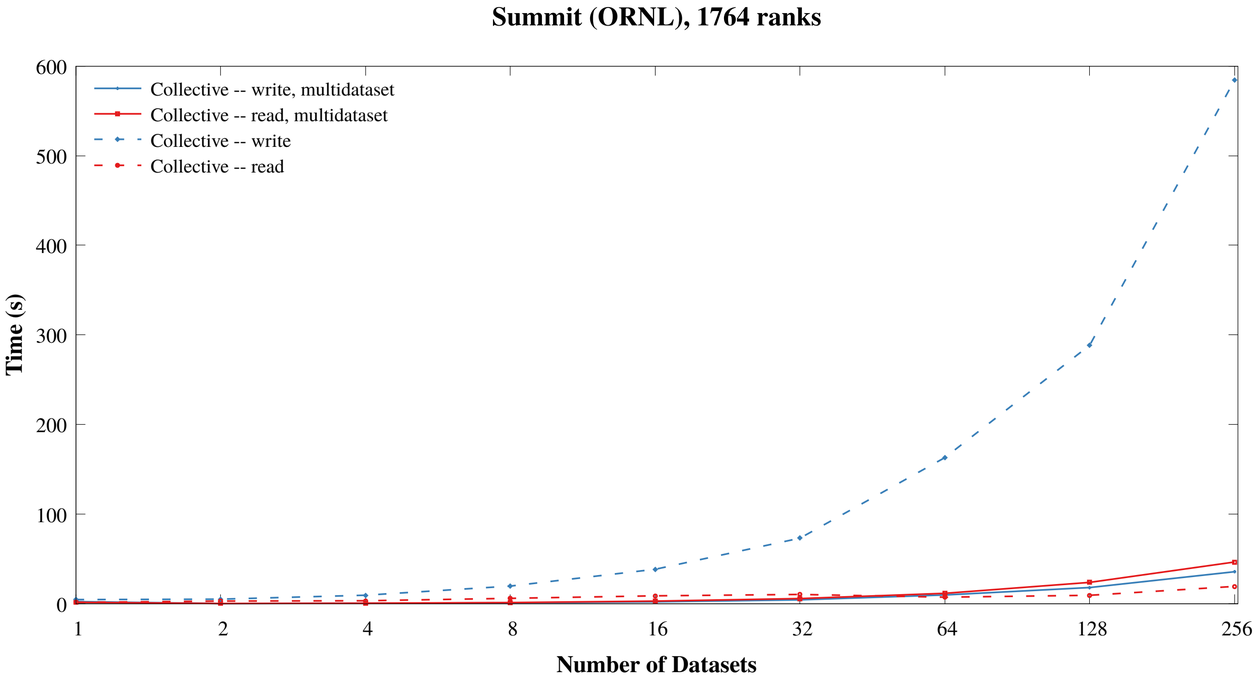
- **Many applications perform I/O on multiple datasets**
- **Current API requires app to issue these I/O calls one dataset at a time**
- **Taking the link-chunk concept further, we would like to aggregate I/O requests involving multiple datasets into a single MPI_File_read/write_at(_all) call**
- **Working implementation available in features/multi_dataset branch**
  - Under final review, will be released in 1.13.3

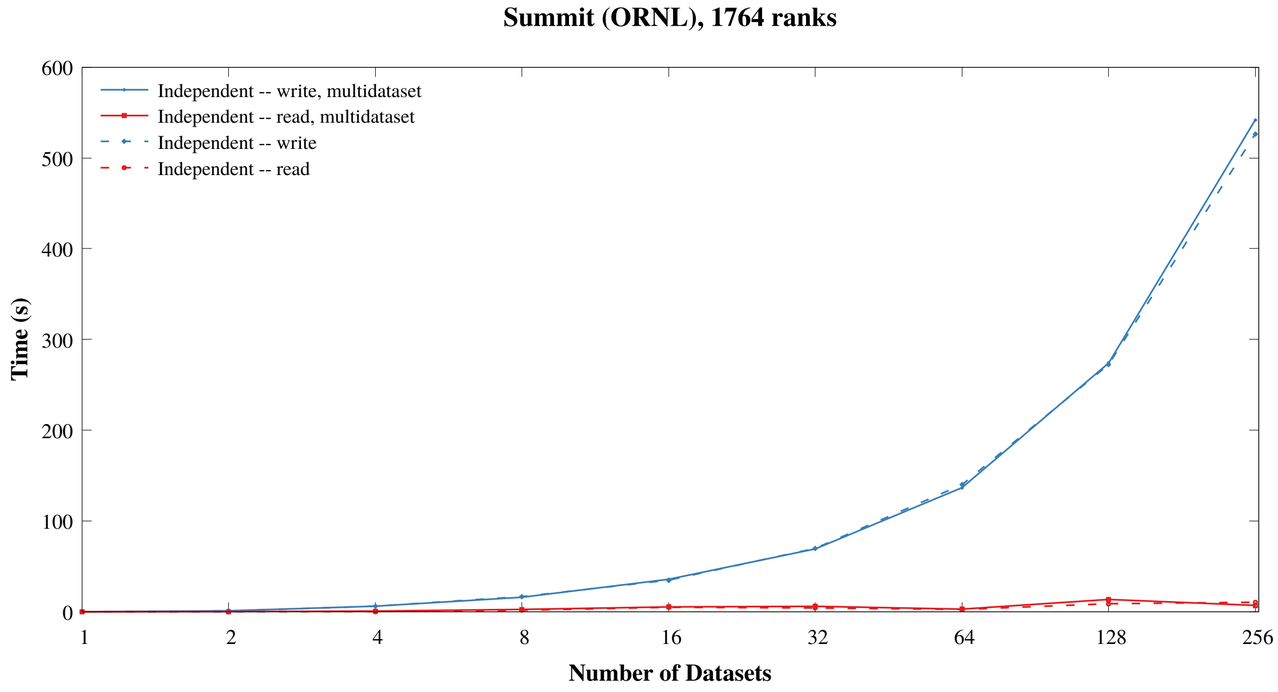
Multi Dataset I/O

- **API:**
  - herr_t H5Dread_multi(hsize_t count, hid_t dataset_id[], hid_t mem_type_id[], hid_t mem_space_id[], hid_t file_space_id[], hid_t xfer_plist_id, void * buf[])
  - herr_t H5Dwrite_multi(hsize_t count, hid_t dataset_id[], hid_t mem_type_id[], hid_t mem_space_id[], hid_t file_space_id[], hid_t xfer_plist_id, const void * buf[])

Benchmark Results

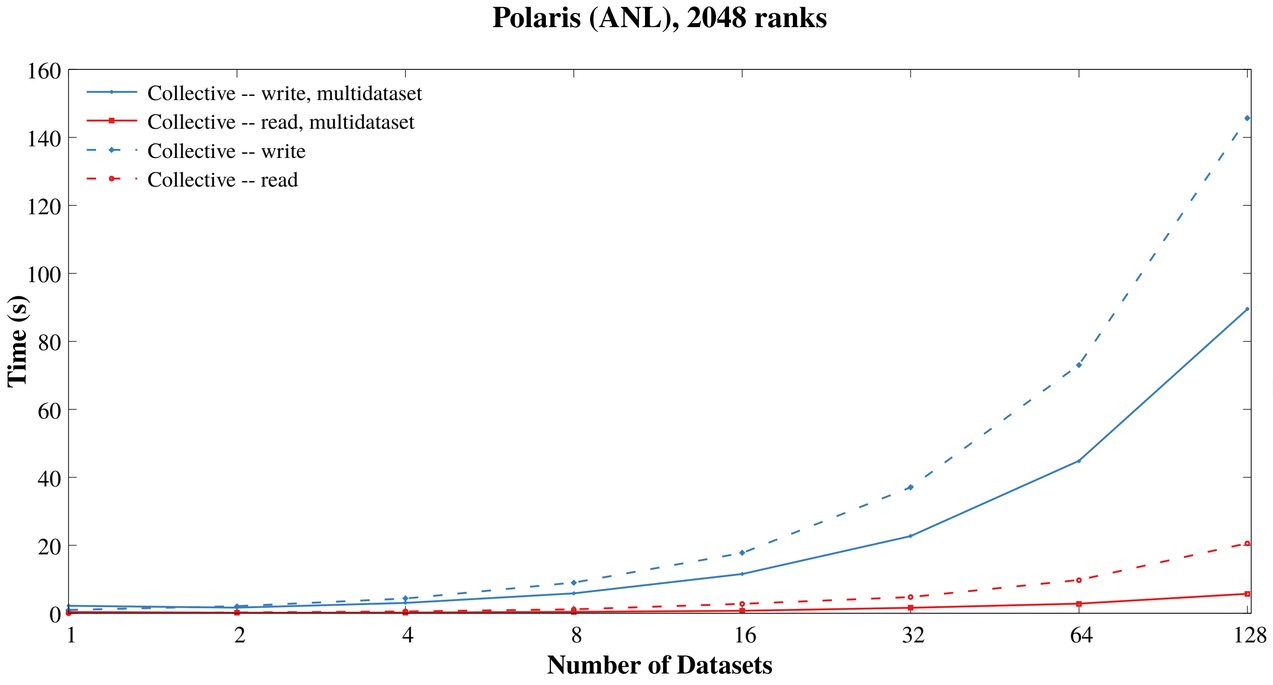
- **Standalone Benchmark**
  - Constant number of ranks, vary number of datasets
  - Compare looped H5Dread/write with H5Dread/write_multi
  - 7 GiB per dataset

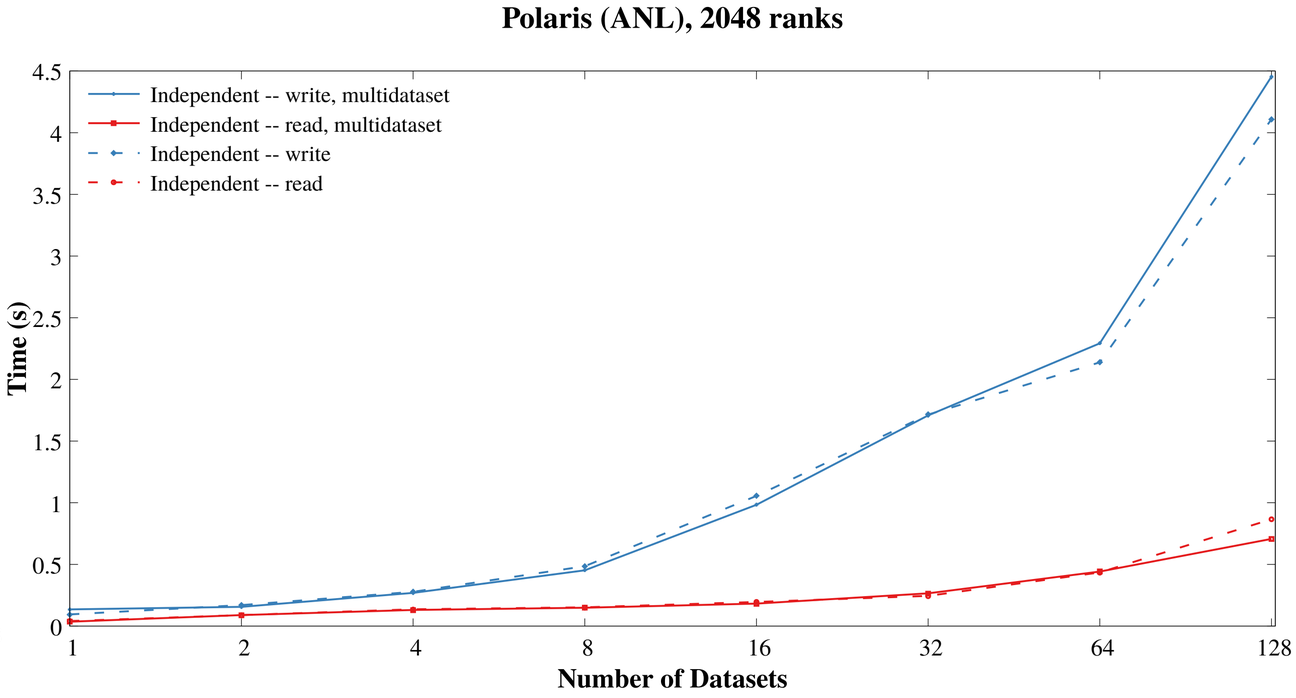




Benchmark Results

- **Standalone Benchmark**
  - Constant number of ranks, vary number of datasets
  - Compare looped H5Dread/write with H5Dread/write_multi
  - 7 GiB per dataset





Benchmark Results

- **Quick CGNS Benchmark**
  - 16 H5Dread/write() calls -> 6 H5Dread/write_multi() calls (don't expect huge improvement)
  - On Summit, with problem size held constant:
    - 2688 ranks, ~10% improvement
    - 10752 ranks, ~6% improvement

Supported Use Cases

- **All ranks must pass the same list of datasets (in collective mode)**
- **All datasets must be in the same file**
- **Each dataset may only be present once in the list**
- **Selection I/O fully supported**
- **For simultaneous multi dataset I/O:**
  - Must be in collective mode – H5Pset_dxpl_mpio
  - None of the datasets can have data filters/compression
  - None of the datasets can involve type conversion
  - All datasets must have contiguous or chunked layout
  - Otherwise, library will process one dataset at a time

Current Status and Future Work

- **Feature branch under final review for integration into mainline develop branch**
  - features/multi_dataset
- **VOL dataset read/write callbacks updated for multi-dataset**
  - Connectors will need to be updated
- **Multi chunk and independent I/O still supported**
  - Link chunk I/O one dataset at a time not supported currently, can implement if there is demand
    - If any datasets in the list cause simultaneous multi dataset I/O to break, other datasets will fall back to multi chunk
- **Plan to implement support for type conversion with collective I/O and multi dataset**

Questions?