Parallel Compression I/O with HDF5; Performance Tuning Techniques

October 25, 2021

M. Scot Breitenfeld
Jordan Henderson
Elena Pourmal

Outline

The HDF Group

- Overview of compression with HDF5
  - Chunking considerations
- Case studies of parallel compression in HDF5

Parallel HDF5 Compression

The HDF Group

- Relatively new functionality introduced in HDF5 version 1.10.2
  - Always seeking feedback on the performance you see in an application

- Limitations
  - ⛔ Doesn’t support independent operations. USE:

```
plist_id = H5Pcreate(H5P_DATASET_XFER);
H5set_dxpl_mpio(plist_id, H5FD_MPIO_COLLECTIVE);
```

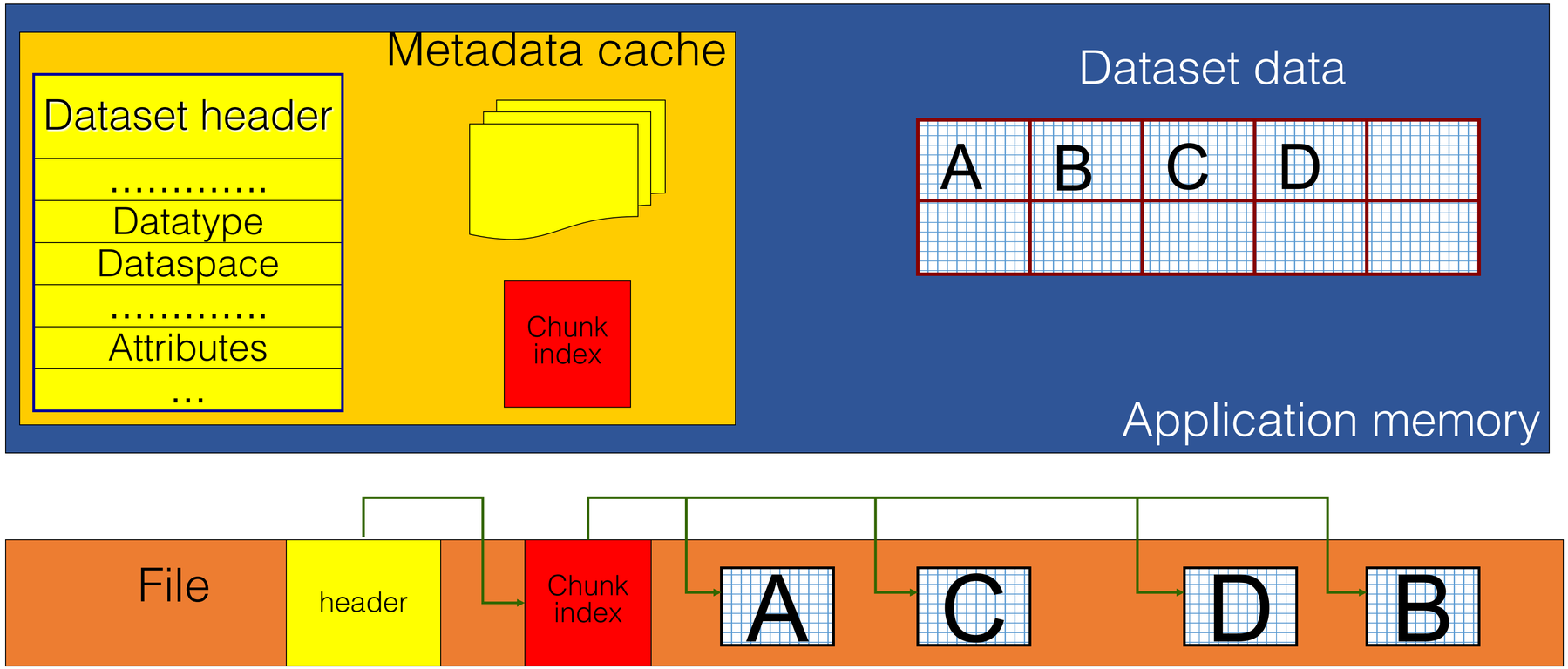
  - Doesn't yet support linked chunk I/O for reading

Chunked Storage

The HDF Group

Good parallel HDF5 compression performance starts with good chunked dataset performance.

- Dataset data is divided into equally sized blocks (chunks).
- Each chunk is stored separately as a contiguous block in HDF5 file.



HDF5 Dataset – Chunked Storage

The HDF Group

- Chunking is required when using extendibility and/or compression and other filters
- **I/O** is always performed **on a whole chunk**
- Understand how **chunking cache** works https://portal.hdfgroup.org/display/HDF5/Chunking+in+HDF5 and consider
  - Do you access the same chunk often?
  - What is the best chunk size (especially when using compression)? NOTE: maximum size for any chunk is 4GB.
  - Do you need to adjust chunk cache size? (1 MB default; can be set up per file or per dataset)
  - `H5Pset_chunk_cache` sets raw data chunk cache parameters for a dataset
    - `H5Pset_chunk_cache (dapl, …);`
  - `H5Pset_cache` sets raw data chunk cache parameters for all datasets in a file
    - `H5Pset_cache (fapl, …);`

HDF5 Dataset – Chunked Storage (cont'd)

The HDF Group

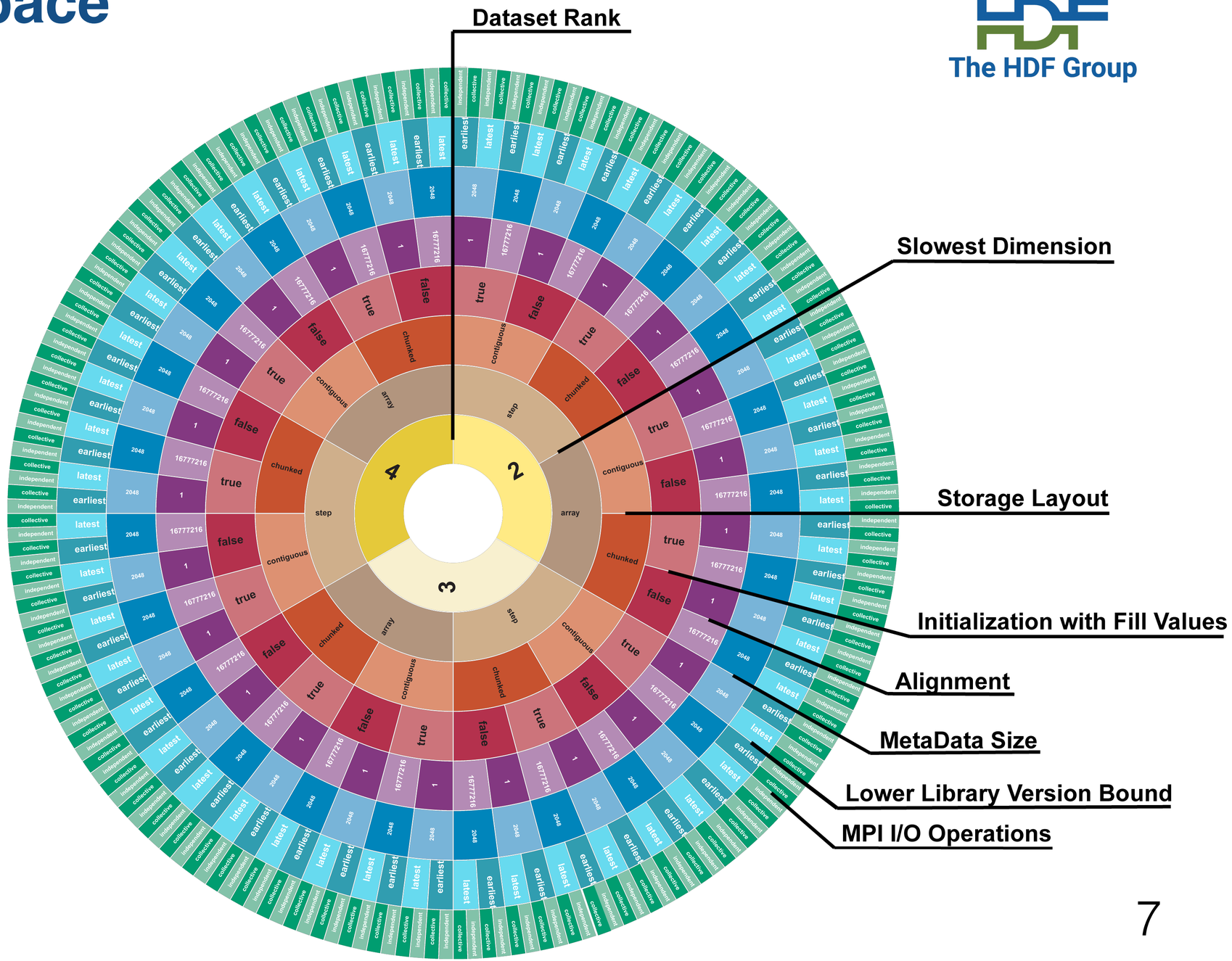
- Cache size is important when doing partial I/O to avoid many I/O operations
- With the 1 MB cache size, a chunk may not fit into cache
  - All writes to the dataset must be immediately written to disk

⚠ **With compression, the entire chunk must be read and rewritten every time a part of the chunk is written to**

  - Data must also be decompressed and recompressed each time
  - Non sequential writes could result in a larger file
- Without compression, the entire chunk must be written when it is first written to the file.
- To write multiple chunks at once increase the cache size to hold more chunks

HDF5 Parameter Space

HDF5 I/O[1] test explores the HDF5 parameter space



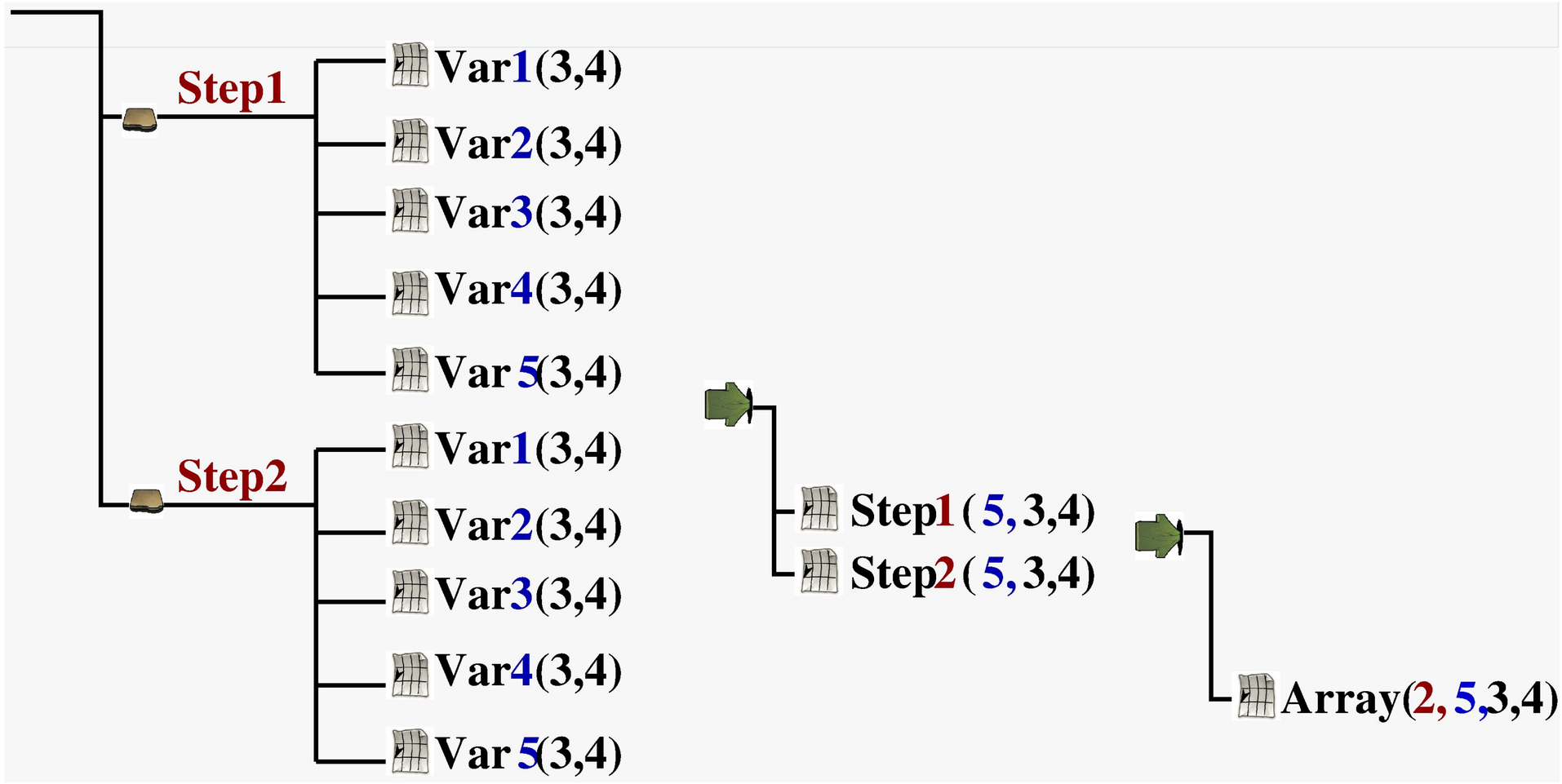
1 https://github.com/HDFGroup/hdf5-iotest

7

IO Pattern Model

The HDF Group

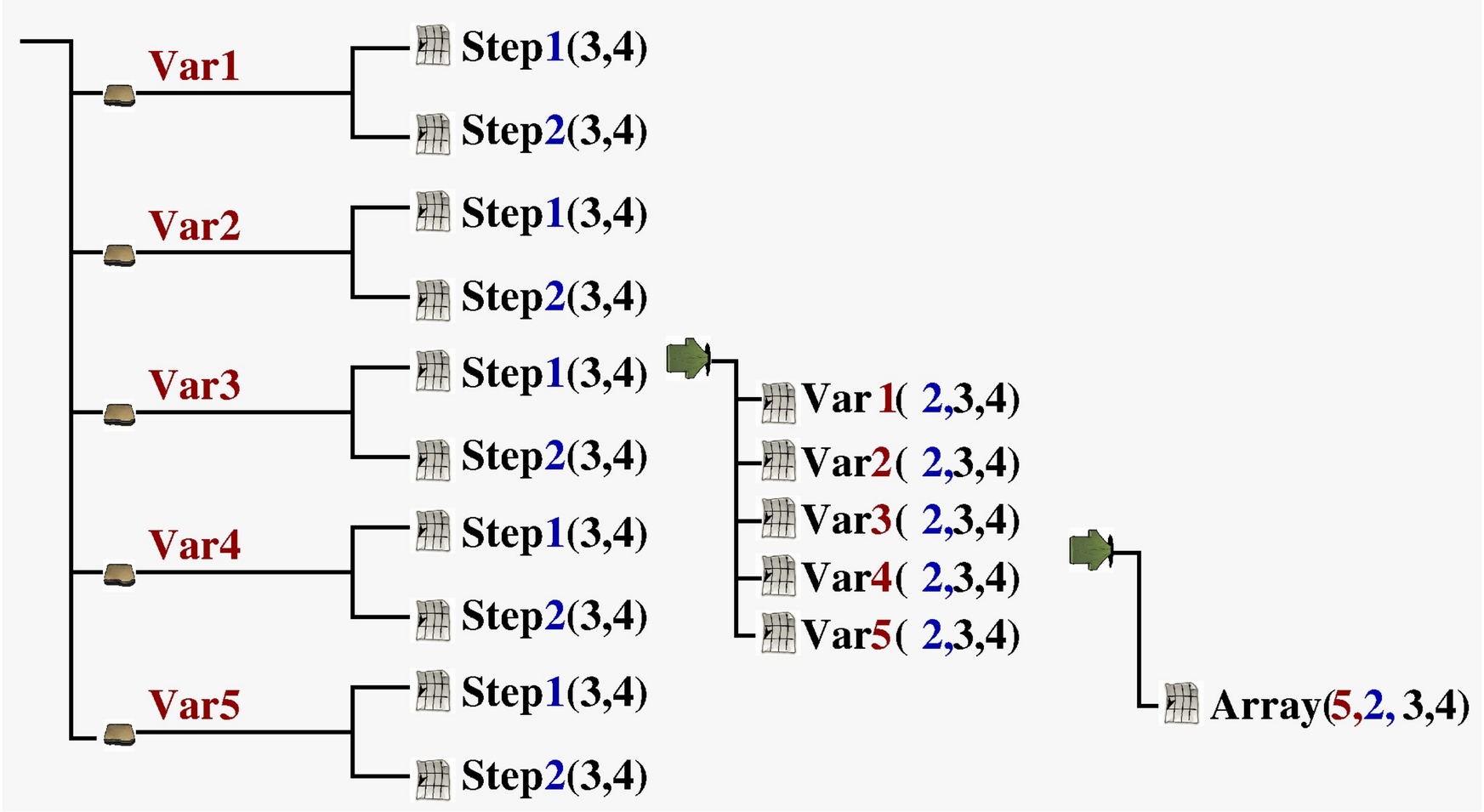
Step based IO Pattern



IO Pattern Model

The HDF Group

Array based IO Pattern



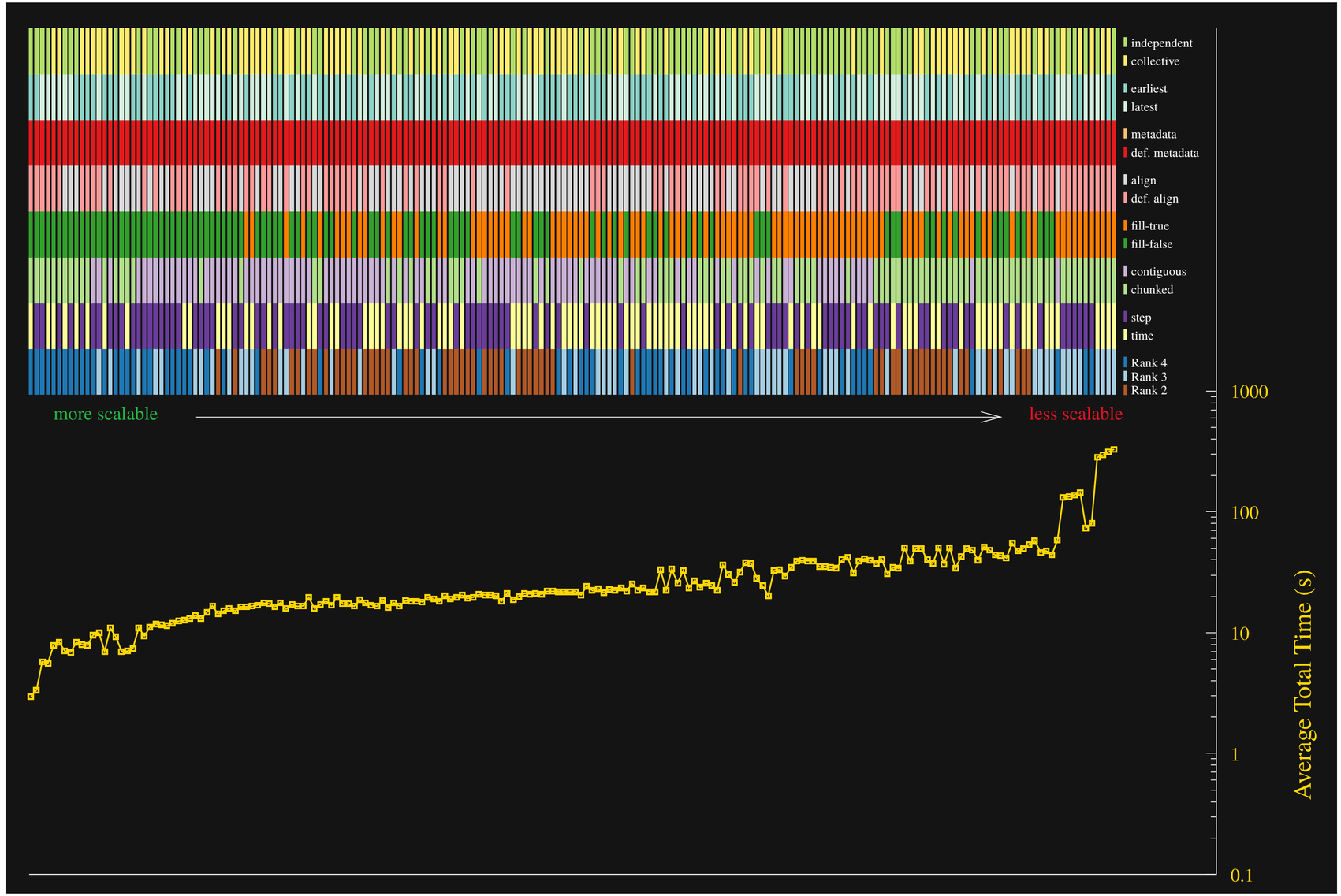
Parallel Compression Case Studies

Parallel HDF5 Compression

The HDF Group

- Start from best weak scaling case scenario from ***hdf5-iotest*** on Summit :

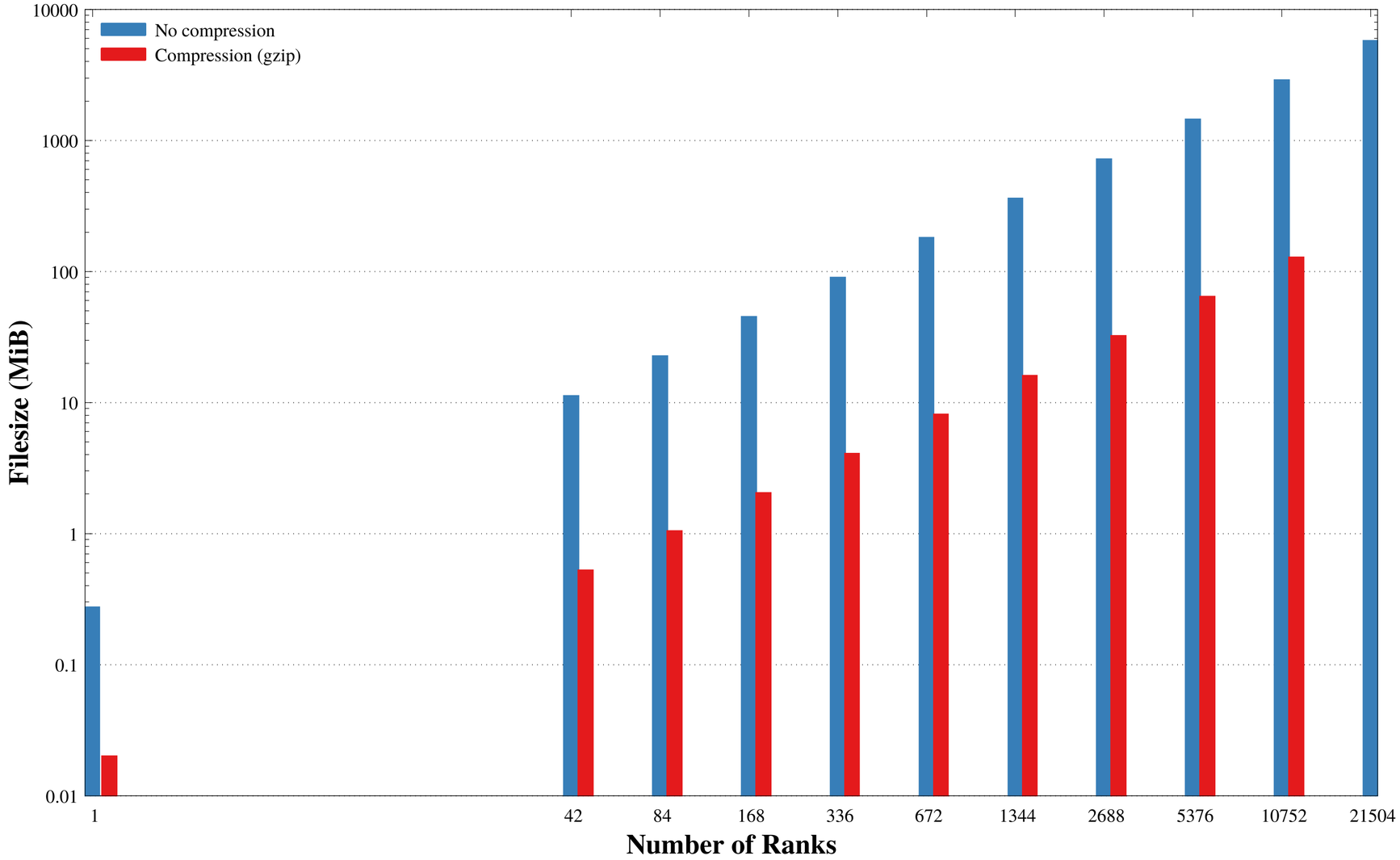
1. Chunked dataset,
2. Collective with no-alignment,
3. Rank four array,
4. Step,
5. No fill values,
6. Earliest library version



Parallel HDF5 Compression

The HDF Group

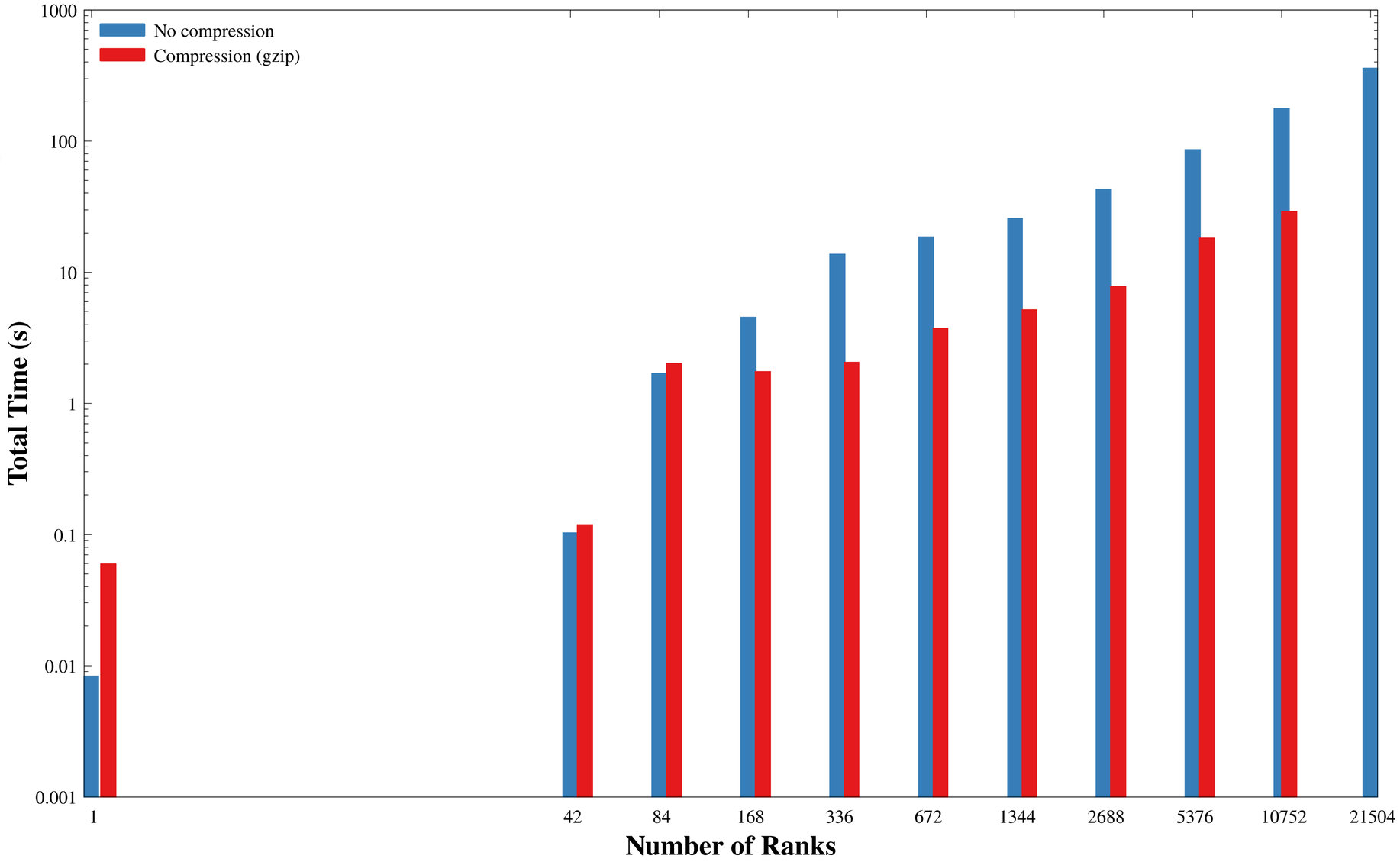
- Compressed and uncompressed file sizes as the number of processes are increased



Parallel HDF5 Compression

The HDF Group

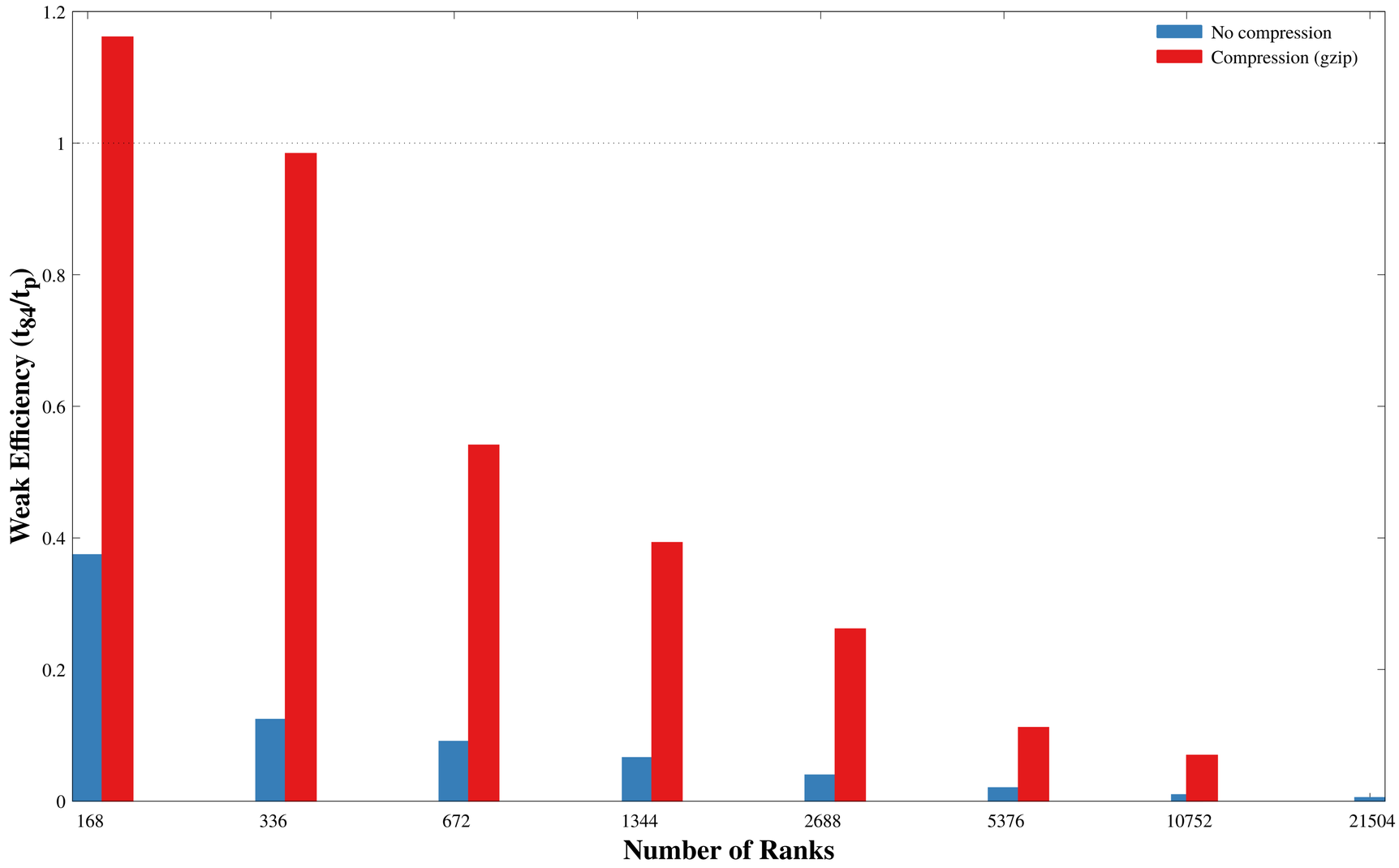
Total time with compression scales proportionally after 168 processes



Parallel HDF5 Compression

The HDF Group

- Weak efficiency relative to 84 processes
- Using compression shows improvement over the non-compression case



CAUTION: Object Creation and Parallel Compression (Collective vs. Single Process)

The HDF Group

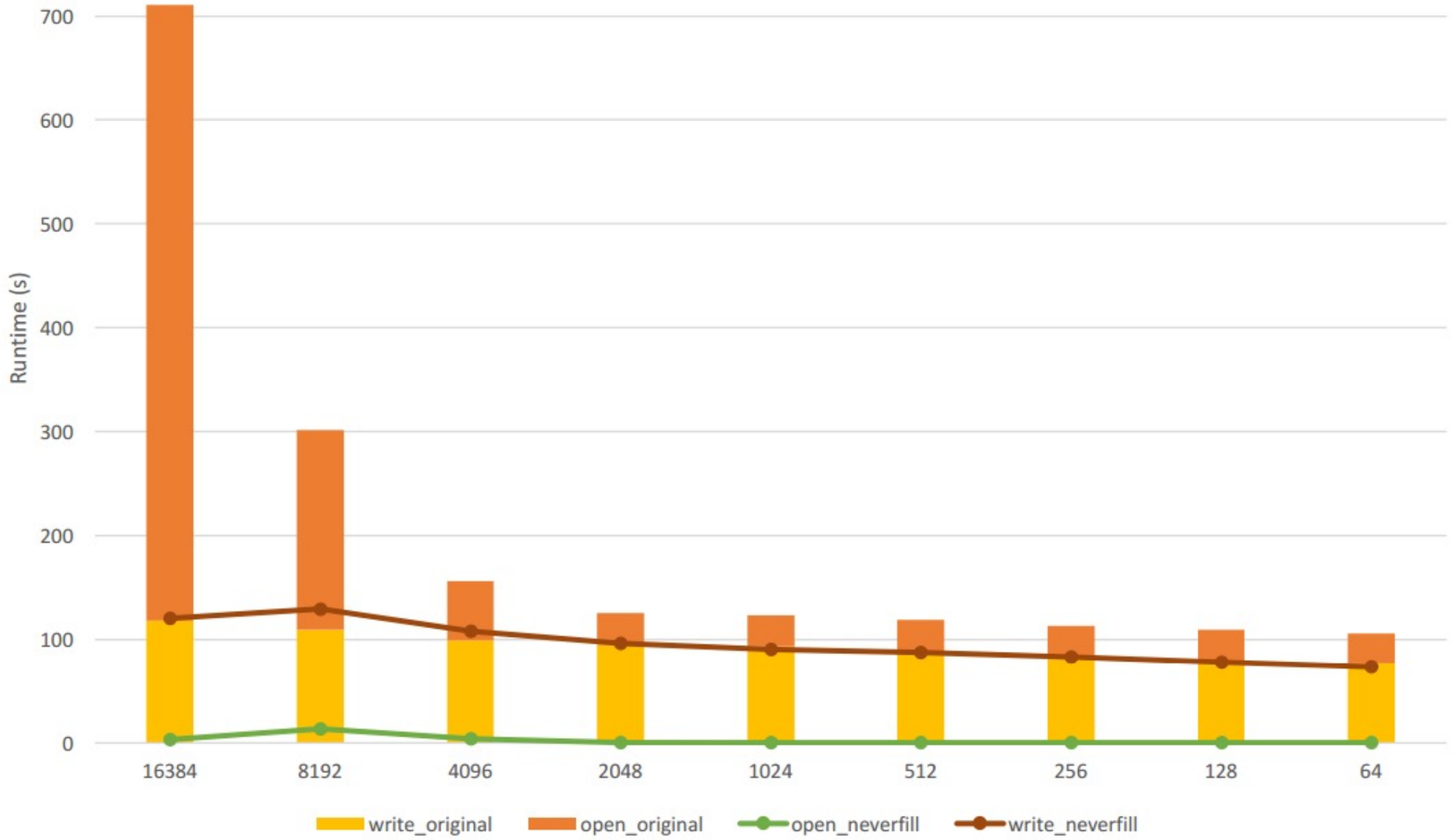
- In **sequential** mode, HDF5 allocates chunks incrementally, i.e., when data is written to a chunk for the first time.
  - Chunk is also initialized with the default or user-provided fill value.

- In the **parallel** case, chunks are always allocated when the dataset is created (not incrementally).
  - This can be an issue if dataset pre-created by rank 0
  - The more ranks there are, the more chunks need to be allocated and initialized/written, which manifests itself as a slowdown

CAUTION: Object Creation (SEISM-IO, Blue Waters—NCSA)

The HDF Group

Set HDF5 to never fill chunks (H5Pset_fill_time with H5D_FILL_TIME_NEVER)



Acknowledgement

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under Award Number DE-AC05-00OR22725.

THANK YOU!

Questions & Comments?