Towards Multi-Thread HDF5

John Mainzer john.mainzer@lifeboat.llc
Elena Pourmal elena.pourmal@lifeboat.llc
Luc Grosheintz-Laval luc.grosheintz-laval@epfl.ch
Matthias Wolf matthias.wolf@epfl.ch

SC22 HDF5 BOF
November 16, 2022

Outline

- Goals
- Concept and prerequisites
- Bypass VOL and current status
- Proof of Concept

Our Goals

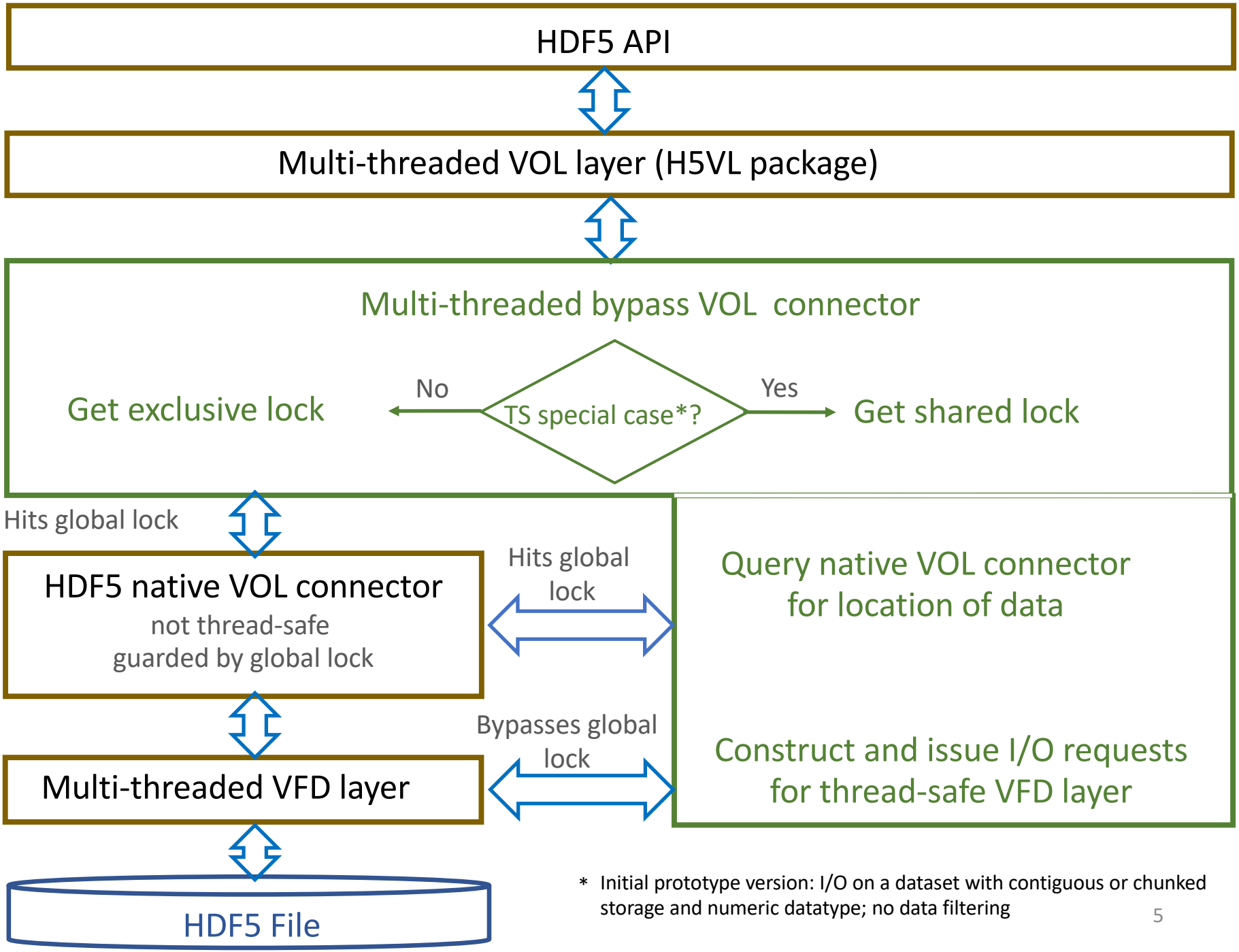
- Design changes to HDF5 required for multi-threaded data access
  - No API changes (some extensions possible)
  - No interruptions to HDF5 library proper and current applications that use thread-safe HDF5
- Enable development of multi-threaded HDF5 VOL connectors and VFDs
- Design and prototype multi-threaded VOL connector to HDF5 storage
- Set path towards full multi-threaded HDF5 implementation

Concept

- Query HDF5 library for the location of raw data
- Execute raw data I/O in parallel in multiple threads

Prerequisites

- HDF5 must allow multiple threads to be simultaneously active in a VOL connector
- For minimal functionality H5E, H5I, H5P, H5CX, and H5VL packages MUST be multi-thread safe
- Other packages are desirable



* Initial prototype version: I/O on a dataset with contiguous or chunked storage and numeric datatype; no data filtering

Bypass VOL connector

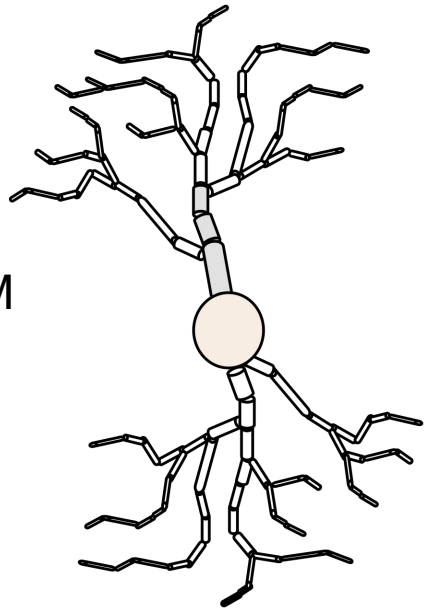
- Examine incoming API calls
- If not a special case, allow only a single thread and route to native VOL
- If special case, query HDF5 for raw data location and execute the specified I/O; multiple threads can run concurrently

Current status

- Design work for the required HDF5 modifications is in progress
- We will make design documents public
- Implementation starts late this year
- Blue Brain developers have implemented algorithm of bypass VOL connector
- And here are the results…☺

Digitally Reconstructed Neurons

1k - 100M neurons



```
{
 "0000": {
  "points”: np.empty((9610, 3), np.float32),
  “offsets": np.empty(21, np.uint64)
 },
 "0001": {
  "points": np.empty((14983, 3), np.float32),
  "offsets": np.empty(48, np.uint64)
 },
 ...
}
```

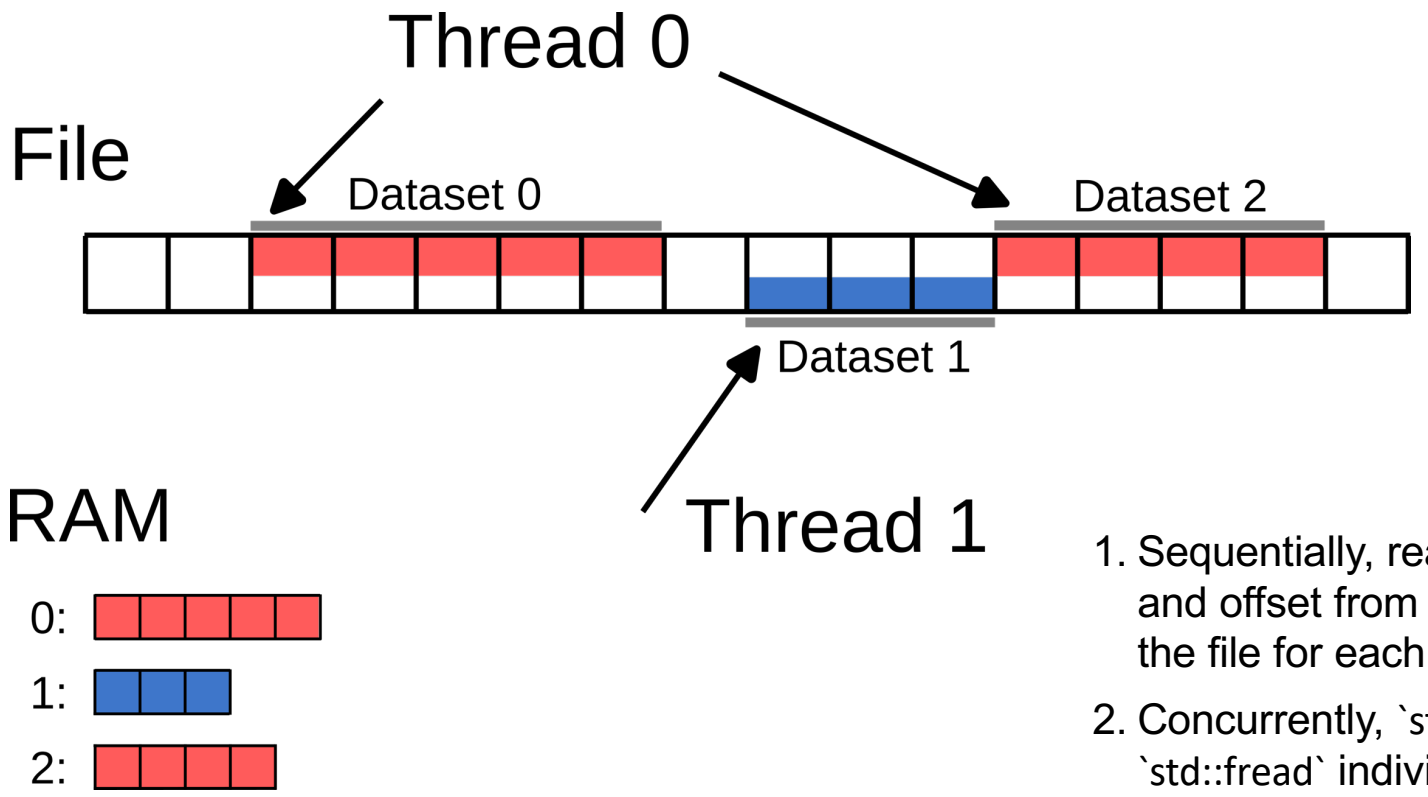
Hardware:

- Intel Xeon Gold 6140
  - 2x 18 cores
  - 6 memory channels
- 100 Gb/s InfiniBand
- SpectreScale/GPFS:
  - 2x GS14KX
  - 8x EDR
  - HDD

Synthetic Data Presented:

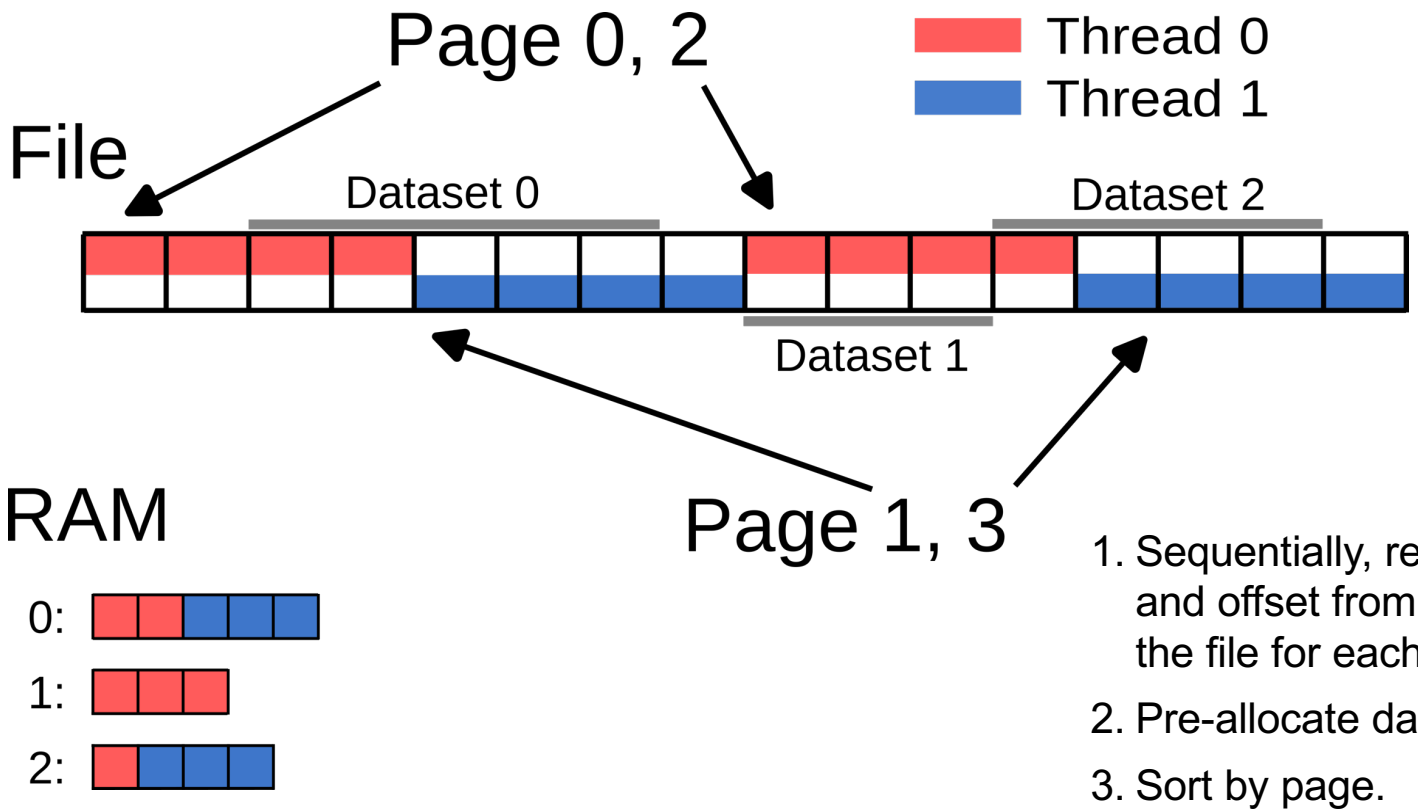
Datasets: 20'000
Total size: 17 GB
File Space Strategy: Page
Page size: 64 kB

Direct OpenMP HDF5 Prototype



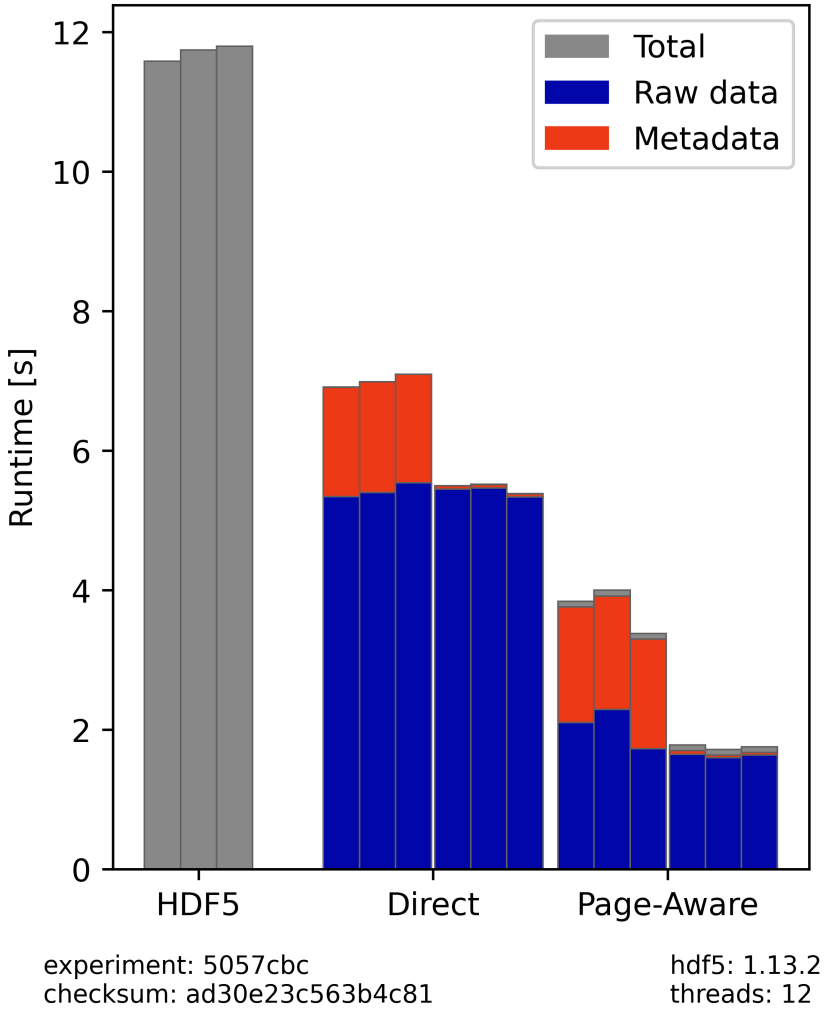
1. Sequentially, read shape/size and offset from beginning of the file for each dataset.
2. Concurrently, `std::fseek` & `std::fread` individual datasets.

Page-aware OpenMP HDF5 Prototype



1. Sequentially, read shape/size and offset from beginning of the file for each dataset.
2. Pre-allocate datasets.
3. Sort by page.
4. Concurrently loop over pages.

Results HDF5 Prototype



Experimental Setup:

- 12 Threads
- 3 measurements
- checksums [1] for correctness

[1]: https://github.com/Cyan4973/xxHash

HDF5: Plain HDF5 with 512 MB page buffer, 75% reserved for raw data.

Direct / Page-Aware: The two variants of the prototype.

- **Left:** Read metadata using HDF5
- **Right:** Read metadata from JSON

Best result **achieves the effective single node bandwidth** of GPFS over InfiniBand.

References

- RFC: Multi-thread HDF5
  https://docs.hdfgroup.org/hdf5/rfc/RFC_multi_thread.pdf
- Blue Brain Project @ EPFL
  https://www.epfl.ch/research/domains/bluebrain/

Acknowledgments

- Material on multi-thread HDF5 is based upon work supported by the U.S. Department of Energy, Office of Science under Award Number DE-SC0022506
- The HDF Group
- Quincey Koziol, Amazon