SPARSE DATA MANAGEMENT IN HDF5

GERD HEBER,THE HDF GROUP

Created: 2019-10-23 Wed 12:53

MOTIVATION

- There's plenty of sparse data
- No concept of sparseness in the HDF5 data model
- Existing facilities can be (ab-)used, e.g.,
  - Chunked layout + compression
  - Mimic a sparse format (RCS, CCS, etc.)
- Common side-effects
  - Hard to determine defined entries
  - Does not preserve array abstraction
- Ad hoc vs. generally applicable solution

REQUIREMENTS

1. Preserve the abstraction
2. Access via the existing HDF5 API
3. Achieve reduction in storage space and I/O time
4. Support random-access R/W operations
5. Support data-parallel operations

DESIGN OPTIONS

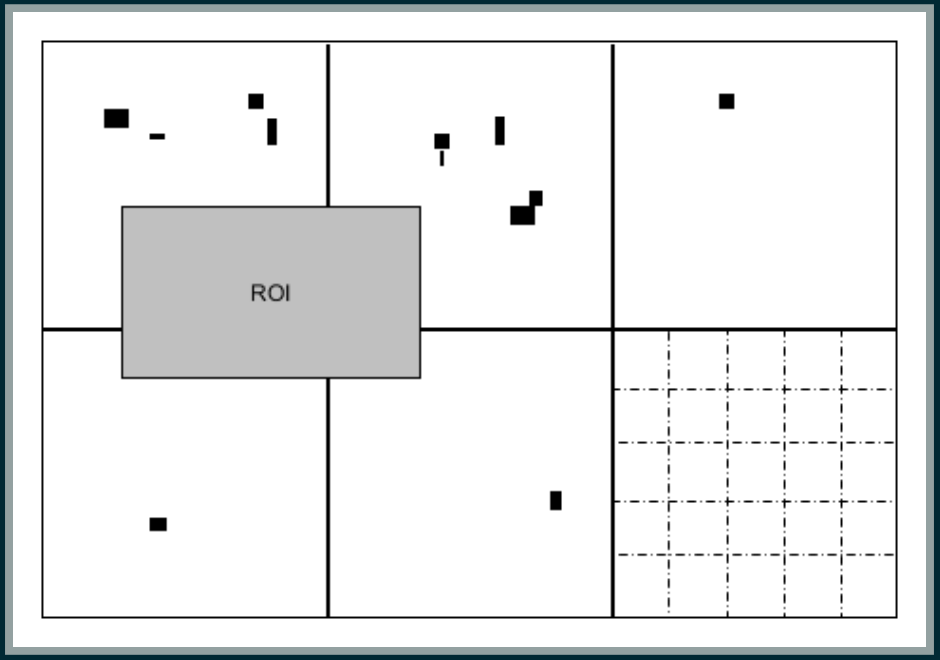
1. Weak options: stretch the existing design
    - Status quo
    - Fill-value filter
    - HL-library
2. Strong options: treat as fundamentally different
    - B-trees
    - Sparse chunks
    - Start from scratch

PROS AND CONS

Approach	Pros	Cons
Status quo	-	Not too sparse
Filter	Space	Memory, structure
HL-lib	Space/time	Destroys abstraction
B-tree	Space/time	Parallel
Sparse chunks	Space/time	Complex selections
From scratch	Perfect	Risk, uncertainty

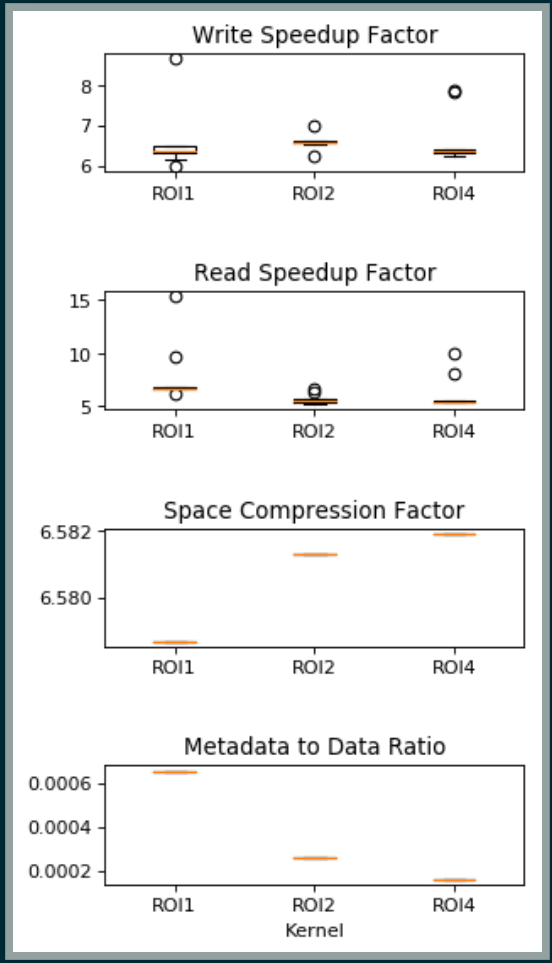
EXPERIMENTS

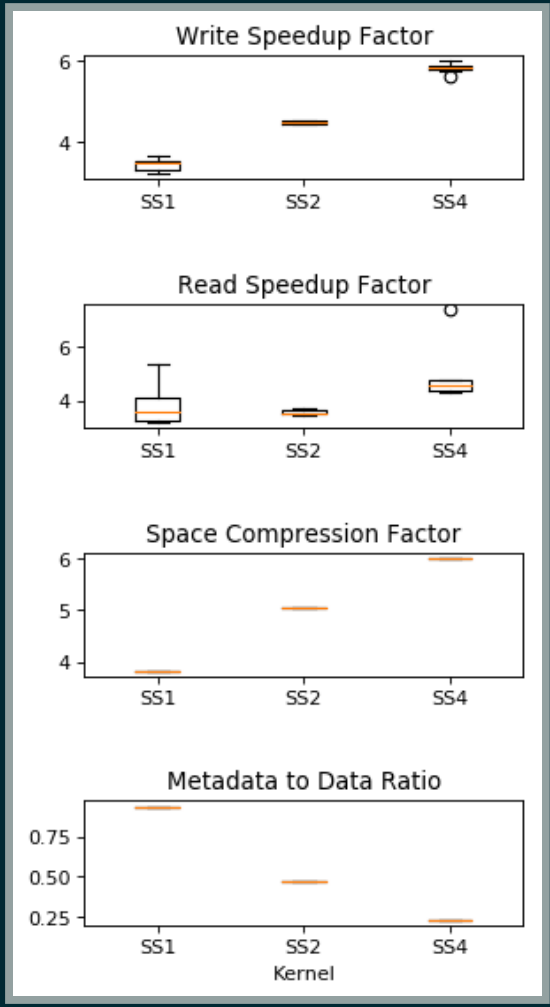
Sample Problem



- Image stream
- Data reduction
  - Reduced number of full frames
  - Write only region of interest or pixel clusters
- I/O kernel(s) to W/R data
- Metrics
  - Baseline: full frames
  - W/R speedup factor
  - Space compression
  - MD/data ratio

SAMPLE RESULTS





NEXT STEPS

- Send us your use cases!
- Paper at XLOOP 2019
- RFC forthcoming
- Community engagement

THANKS

- DOE EOD-HDF5 Project (Dr. Laura Biven)
- LBNL (Quincey Koziol, Suren Byna)
- FNAL (Mark Paterno)
- Steven Varga ("Mr. H5CPP")
- The HDF Group (John Mainzer, Neil Fortner, Elena Pourmal)

Created by gheber.