Efficient Trace Storage and Access

Vincent Immler
vincent.immler@oregonstate.edu

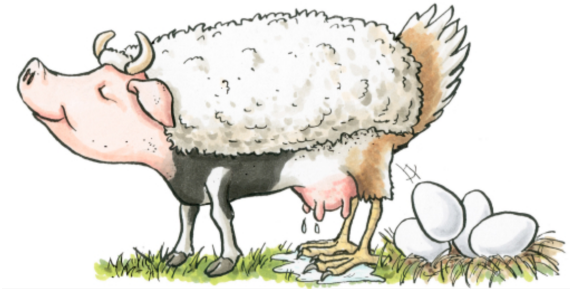
OPTIMIST 2024, October 24, 2024

OPTIMIST 2024

Oregon State University

rootoftrust.io

What We Seek ...



German folklore creature “die Eierlegende Wollmilchsau”

- ... *may not exist*: some features may be incompatible
- Some features may take time [1]
- At the end of the day: work needs to be done (choose what works best in your situation)

[1] Coroutine RFC for Rust: a mind-boggling 7 years without progress: `https://josephg.com/blog/rewriting-rust/`

Interoperability and Features vs. Efficiency

- With interoperability comes a lot of fluff
- Interoperability can lead to standards such as ASN.1 (parsing is fun!?)
- Think of this is a spider chart: what are the relevant criteria to assess
- Perhaps: focus on terminology / uniform key words, less so on API
  - First step to come up with an API / standard anyway
  - Leave interoperability as implementation “exercise”
- MANY different scenarios: power, EM, multiple probes, frequency/phase, …

Feature: Contiguous vs. Chunked File Formats

Contiguous file format: flattened array of data

- Pro: fastest for full sequential read of data
- Con: slow if slices of data are needed (i.e., indexing)

Chunked file format: chunks of data

- Pro: slices faster, easily works with compression
- Con: slower full sequential read; expensive re-chunking

Examples:

- .trs, .ets, .npy: not chunked
- .h5: contiguous (default), chunked (when compressed) – single file
- .zarr: chunked (default) – sparse storage (multiple files)

→ cannot have both at the same time!

Feature: Compression

Why Care? "Storage is Cheap!"

- Storage might be cheap, but it is not for free either
- Equity aspects for teaching: students with laptops and limited storage
- Economic and ecologic aspects: why waste a resource

Less Data = Less I/O = Faster!

- Compression is the future as I/O bandwidths fail to keep up with compute
- Especially important for clustered computing to reduce load on HPC fabric
- lz4: "Designed to transmit data to the processor cache faster than a memcpy() OS call"
- SCA data too noisy? Thanks to 12-bit ADC, 4 bit (25%) compression for free!

→ Compressed data sets should be the default; uncompressed the rare exception

Proposed File Format for the Academic Community: Zarr

Reasons

- Chunked file format; v3 with ‘shards’ (a superset for chunks)
- Supports compression including lz4 and lz4hc
- Sparse file format, great for cloud computing (e.g., Dask)
- True parallel-read
- Lightweight instead of monolithic design (i.e., unlike HDF5)
- Community-governed
- Likely superior with future improvements, such as `io_uring`
- Data should be LSB-aligned (regardless of 10-bit or 12-bit ADC)

Key Words and Structure (work-in-progress)

`/X/Y/...`

- Shall we call it /X/Y/traces? or Samples?
- How shall we handle optional data (such as masking values)?
- /X/Y/metadata with specific structure? Or free to choose?
- …

Just this first step alone might require a lot of work already (prior to defining an API)

Thank you for your attention!
Questions?

OPTIMIST 2024

vincent.immler@oregonstate.edu