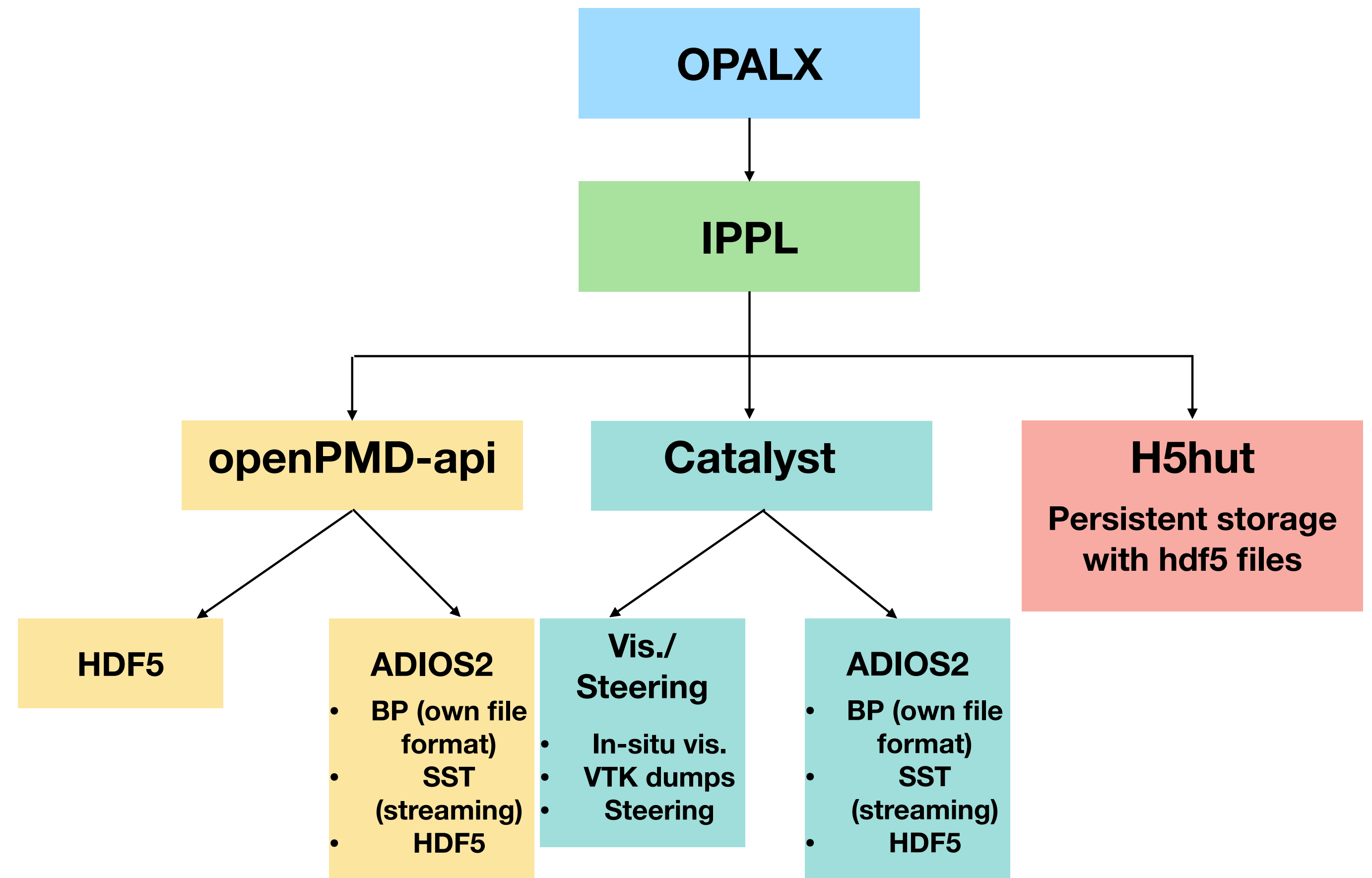


I/O for OPALX (Sri, Severin, John)

Goals for the retreat

1. With a minimal IPPL example (e.g. TestScatter) investigate three possible routes for the persistent I/O in IPPL/OPALX
2. **H5hut**: The legacy option in OPAL (John/Andreas)
3. **Catalyst->ADIOS2->HDF5/BP**: Based on the existing catalyst in-situ interface in IPPL (Severin)
4. **openPMD-api->ADIOS2->HDF5/BP**: Interface openPMD-api with IPPL (Sri)
5. We are interested in: Performance, scalability, dependencies and maintenance



OpenPMD interface with IPPL

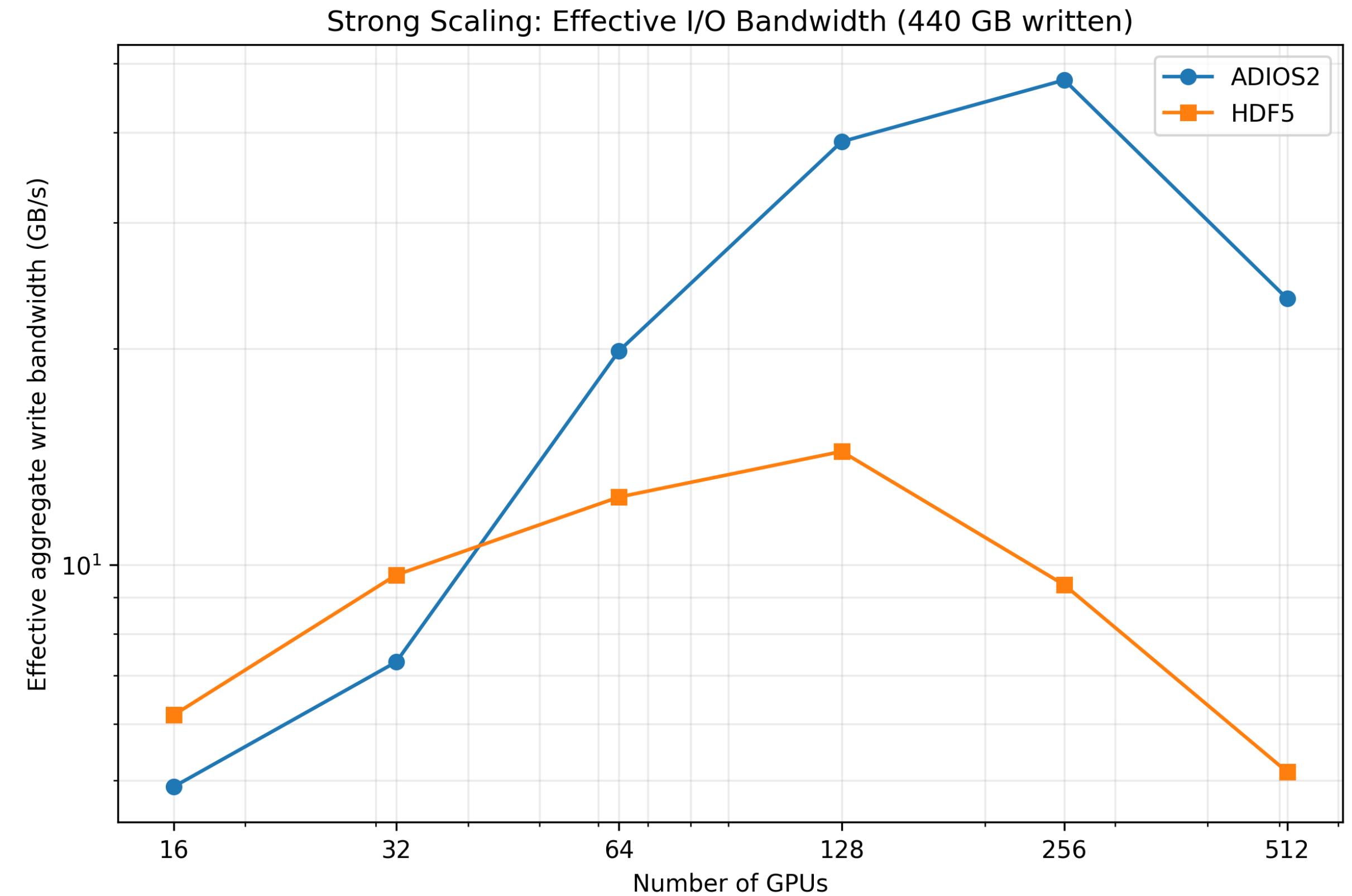
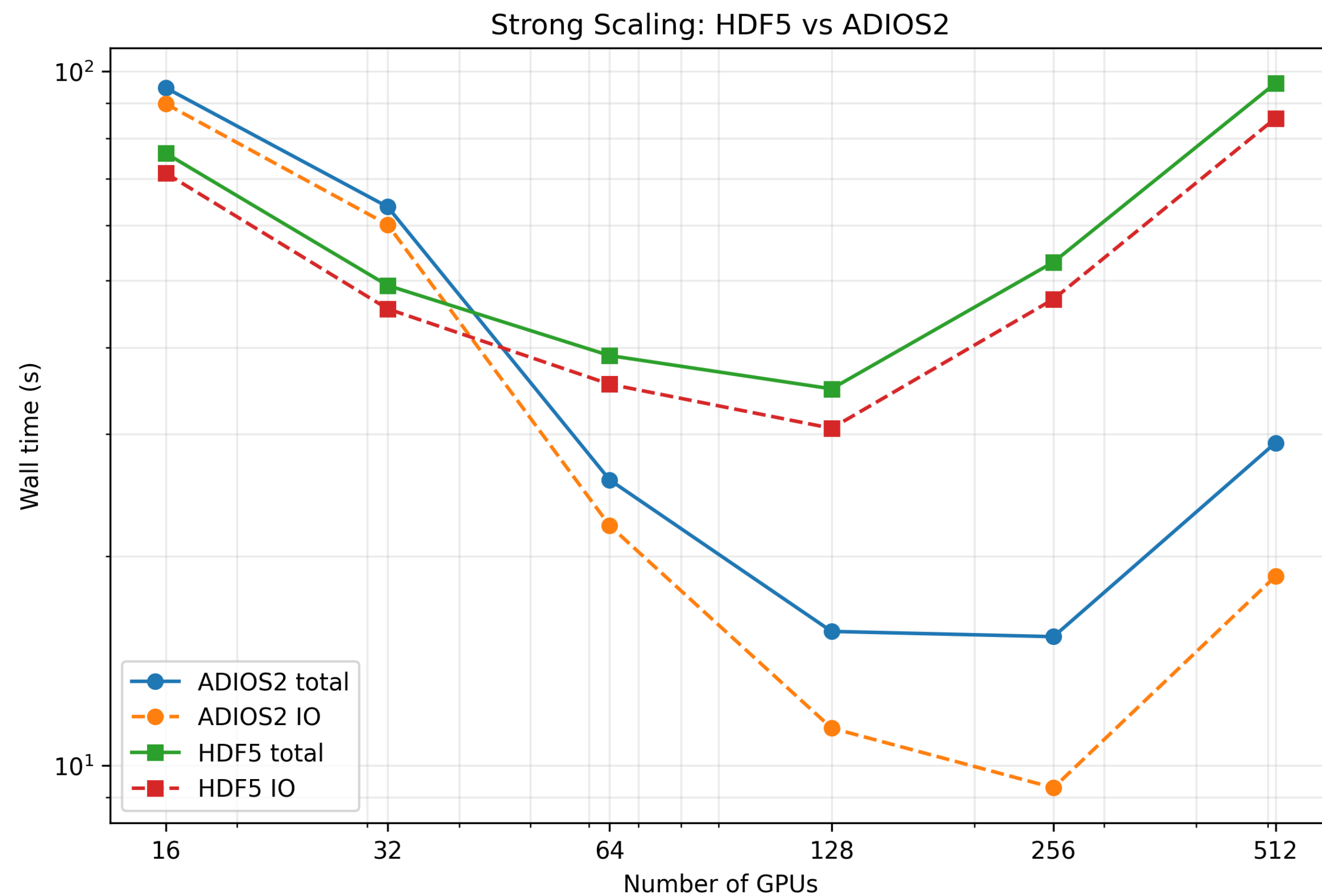
- OpenPMD-api was interfaced with IPPL through FetchContent
- HDF5 and ADIOS2: used through pre-installed modules. FetchContent also possible but somewhat heavy
- Writes out an hierarchical format like
 - data/0
 - data/0/fields
 - data/0/fields/charge_density
 - data/0/fields/charge_density/SCALAR
 - data/0/particles
 - data/0/particles/bunch
 - data/0/particles/bunch/position
 - data/0/particles/bunch/position/x
 - data/0/particles/bunch/position/y
 - data/0/particles/bunch/position/z
 - data/0/particles/bunch/positionOffset
 - data/0/particles/bunch/positionOffset/x
 - data/0/particles/bunch/positionOffset/y
 - data/0/particles/bunch/positionOffset/z
 - data/0/particles/bunch/weighting
- Correctness verified through h5py, Paraview. OpenPMD-viewer interpreted the field as vector but probably can be fixed

OpenPMD interface with IPPL

- Requires struct of arrays layout for vector particle attributes/fields-
>Data copy (but this seems to be the case with H5hut too!)
- Direct passing of device pointers failed: But this is due to HDF5 or ADIOS2 installation (John looking into it) and not the restriction of OpenPMD-api itself
- HostView worked with both HDF5 and ADIOS2. Seems OpenPMD-api directly supports Kokkos views too (need to look at it)

Benchmark

- TestScatter with 512^3 grid, 10 particles per cell (~1 billion particles). Same data written for 10 iterations. 44 GB data/time step. Total 440GB. JUWELS Booster A100 GPUs



- ADIOS2 (bp5) scales much better than HDF5 at high ranks. ~4-5X higher bandwidth achieved than HDF5 at scale. Need to do further tests/benchmarks

Catalyst/Conduit Route

- Conduit->ADIOS->bp5 or h5: Conduit has direct support only for ADIOS1 and not ADIOS2
- Catalyst->ADIOS2->bp5 or h5: Supports only non-adaptive mesh-based data and no particles
- Conclusion: Use Conduit/Catalyst route only for in-situ visualization/steering (as it is meant) and not for persistent I/O

H5hut or OpenPMD-api for persistent I/O?

- H5hut: Quick inspection shows we also need **data copy** for AoS->SoA. Benchmarks need to be done (may be move to IPPL to be consistent with other I/O?). But performance may be mostly similar to OpenPMD->HDF5 route
- Since OpenPMD-api in addition supports ADIOS2 (supports streaming, bp5 etc.) it is worth considering
- **Conclusion: May be move H5hut interface from OPALX to IPPL and due to similarity of interfaces could switch to H5hut or OpenPMD-api at compile time. If we gain enough confidence (or not) on openpmd in future can use only one of them?**