HPC and data management for HEP

SciDAC HEP Data Analytics on HPC

N.Buchanan[2], S. Calvez[2], P. Carns[1], P.F. Ding[4], M. Dorier[1], D.Doyle[2], A. Himmel[4], J. Kowakowski[4], R. Latham[1], M. Paterno[4], A. Norman[4], S. Sehrish[4], A. Sousa[3], S. Snyder[1], R. Ross[1]

[1]Argonne National Laboratory, [2]Colorado State University, [3]University of Cincinnati, [4]Fermi National Accelerator Laboratory

FERMILAB-POSTER-18-097-CD

NOvA Event Selection

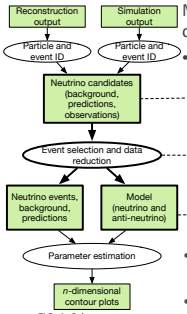

FIG. 1: Science context and traditional solution

Measurement of the neutrino oscillation parameters by the NOvA collaboration, PRL 118, 231801 (2017).

- Sample of ~27 million reconstructed spills to search for electron-neutrino appearance events.
- Events are stored in a ROOT n-tuple format.
- ~180 thousand ROOT files.
- File sizes range from a few hundred KiB to a few MiB; the full dataset is ~3 TiB.
- Strict event selection criteria to reduce millions of theses events to a few tens.
- Then summarize these tens of events to be used the extraction of the neutrino oscillation parameters.
- Goal: ***reduce the time it takes to process analysis-level data.***

HEP Data management

- Experimental HEP deals with complex Big Data from unique detectors.
  - Already ~10s of PB for CMS at US centers, expected to grow by 2 orders of magnitude during HL-LHC era
  - ~17 PB/year for DUNE
- Analysis tasks and therefore analysis software is also complex.
- The analysis software is written by thousands of physicists .
- We provide a modular framework and data model to support their collaboration.
- ***We are exploring the design of an object model that leverages HPC machine features.***

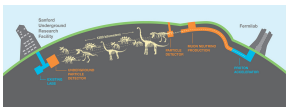

FIG. 5: DUNE

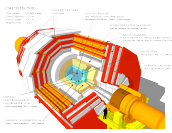

FIG. 6: CMS detector

Using HPC for Parallel Event Selection

Most of the computing resources that will be available to us will be at HPC centers.

- Modify our workflows and code to take full advantage of HPC systems.
- Minimize *reading*, *communication* and *synchronization* between processes, parallelism is *implicit*; user-written code looks just like serial code.
- Process all data for a given *slice* in a single MPI rank, the *slice* is NOvA's "atomic" unit of processing, like a collider *event*.
- Organize the data into a single HDF5 file, containing many different tables.

Example of new-style data organization in HDF5

run	subRun	event	slice	distallpngtop	...35 more...
16433	61	356124	35	nan	
16433	61	356124	36	-0.74013746	
16433	61	356124	37	nan	
16433	61	356125	1	nan	
16433	61	356125	2	423.6337	
16433	61	356125	3	-2.849864	

Table 1. This table has one entry per *slice*. We are doing *slice-by-slice* selection.

run	subRun	event	slice	vtxid	npng3d	...6 more...
16433	61	356124	35	0	0	
16433	61	356124	36	0	1	
16433	61	356124	36	1	1	
16433	61	356124	36	2	5	
16433	61	356125	1	0	1	
16433	61	356125	3	0	0	

Table 2. This table has one entry per *vertex*. Some slices have none (and do not appear in this table). Some slices have many vertices.

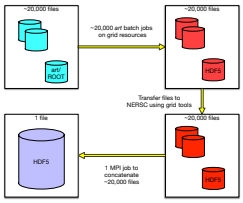

FIG. 2: Workflow for translating old-style data to new-style data

Distributing and reading information

- Each rank reads its "fair share" of index info from each table.
  - identifies which rank should handle which event, for most even balance
  - identifies range of rows in table that correspond to each event (all slices)
- Event "ownership" information distributed to all ranks.
  - this assures no further communication between ranks is needed while evaluating the selection. criteria on a slice-by-slice basis.
  - perfect data parallelism in running all selection code.
- Each rank reads *only* relevant rows of relevant columns from relevant tables.
  - all relevant data read by some rank.
  - no rank reads the same data as another.

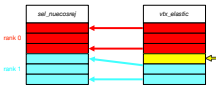

FIG. 3: An example of distributing and reading

User code Examples

Listing 1. Selection on multiple columns of a table

```
def kNueSecondAnaContainment(tables):
    df = tables['sel_nuecosrej']
    return (df.distallpngtop > 63.0) & \
           (df.distallpngbottom > 12.0) & \
           (df.distallpngeast > 12.0) & \
           (df.distallpngwest > 12.0) & \
           (df.distallpngfront > 18.0) & \
           (df.distallpngback > 18.0)
```

Listing 2. Groupby and Aggregation example

```
def vtxelasticyCut(tables):
    df = tables['vtx_elastic']
    df['good'] =
        (df.vtxid == 0) & (df.npng3d > 0)
KL = ['run', 'subRun', 'event', 'slice']
return df.groupby(KL)['good'].agg(np.any)
```

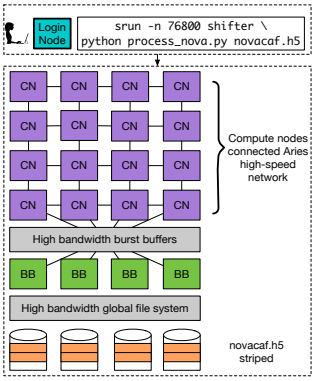

FIG. 4: HPC solution

Using Object store for HEP data HEPnOS

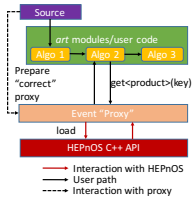

FIG. 7: User interaction with HEPnOS

HEP data organization

- A common design is to have levels of data aggregation: data set / run / subrun / event / data products
- At each level of aggregation, the named items are independent.
- Data sets can contain other data sets.
- Distribution of data automated by the data model implementation, not user-visible.

Goals

- Manage physics event data from simulation and experiment through multiple phases of analysis
- Accelerate access by retaining data in the system throughout analysis process
- Reuses components from Mochi ASCR R&D project

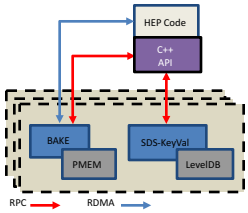

FIG. 8: HEPnOS

Properties

- Write-once, read-many
- Hierarchical namespace (datasets, runs, subruns)
- C++ API (serialization of C++ objects)

Components

- Mercury, Argobots, Margo, SDSKV, BAKE, SSG
- New code: C++ event interface
  Map data model into stores

Demo

- Using existing *art* framework-related software:
  - *gallery* provides access to event data in *art*/ROOT files from outside the *art* framework.
  - pre-existing builds of the LArSoft data product libraries.
- Using some of the data product classes from LArSoft (used by DUNE, *etc.*)
  - `std::vector<recob::Hit>`
  - `std::vector<recob::Track>`
  - `art::Assns<recob::Hit, recob::Track>`, which are bidirectional associations between hits and tracks.
- Using Docker containers for easy portability of development environment
- Prototype test programs are already running to:
  - read from existing *art*/ROOT data files, and to write to the new data store
  - read from the new data store, and verify the integrity of the data

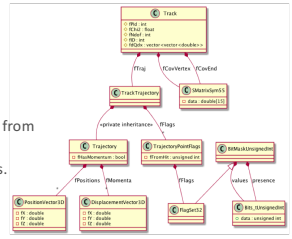

FIG. 9: class recob::Track

Listing 3. Example of HEPnOS use: reading from the object store

```
hepnos::DataStore datastore(configfile);
for (auto it = datastore.begin(); it != datastore.end(); ++it)
  for (auto const& ds : *it)
    for (auto const& r : ds.runs())
      for (auto const& sr : r)
        for (auto const& e : sr) {
          std::vector<recob::Hit> hits;
          art::InputTag hit_tag("gaushit", "", "PrimaryReco");
          e.load(hit_tag, hits);
          // .. do something with the vector of Hits ..
        }
```

The user interface looks a lot like our current user interface

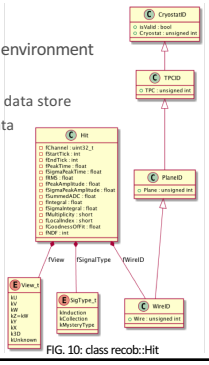

FIG. 10: class recob::Hit

Summary and Future work

- NOvA is taking ownership of our HDF "ntuple" production code.
- We will be doing large-scale performance testing of the code.
  - Similar design processing LArIAT data demonstrate perfect scaling to 76,800 ranks; *read* and *decompress* 42 TB of data in < 20 seconds wall-clock time.
- We will be comparing performance with DIY C++ 14 implementation.
- Integration with larger workflow using Decaf that is also part of the SciDAC project.
  - use of changes in event selection criteria to evaluation systematic uncertainties in the mixing parameter measurements.
  - one integrated MPI program, to take best advantage of HPC platform.

Summary and Future work

- We are exploring how to put support for MPI parallelism directly into the infrastructure.
  - Will enable our reconstruction/analysis codes to take advantage of data parallelism and distributed programming without looking much different than it does today.
- We will deploy to NERSC (through Shifter) and ALCF (through Singularity) and perform scalability study.
- We will be comparing performance against current methods of reading and writing data.
- Integration of object store with larger workflow utilizing Decaf and enable its use at different levels of processing.

Acknowledgement: This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) program, grant 1013935.

2018 Scientific Discovery through Advanced Computing (SciDAC-4) Principle Investigator Meeting, Rockville, MD

http://computing.fnal.gov/hep-on-hpc/