Implicitly Parallel Analysis Tools

Derek Doyle, Micah Groh, Jim Kowalkowski, Andrew Norman, Marc Paterno, Saba Sehrish

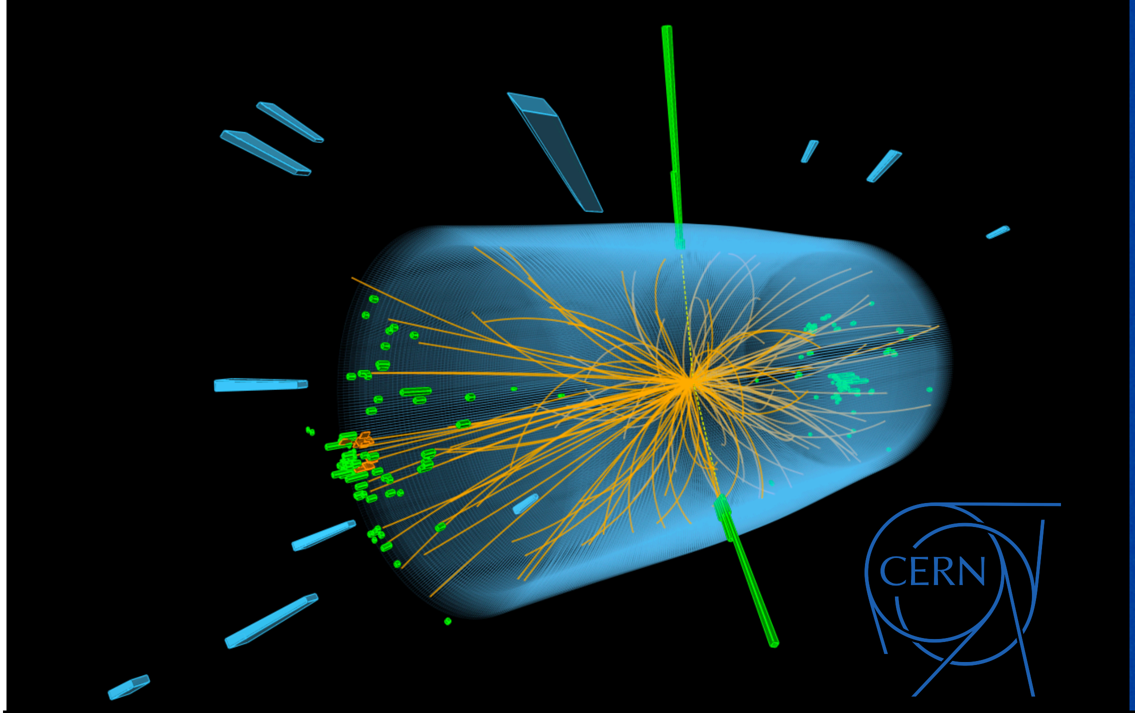
PyHEP 2022

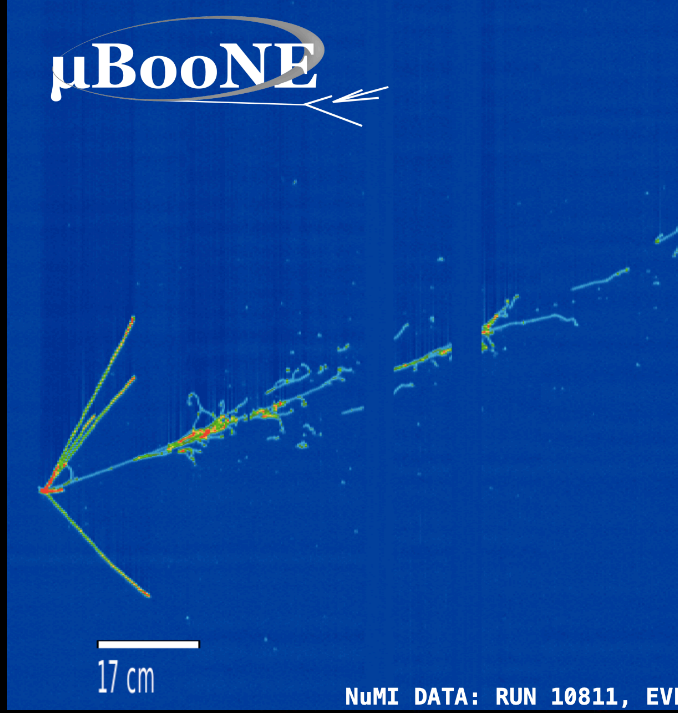
September 15, 2022

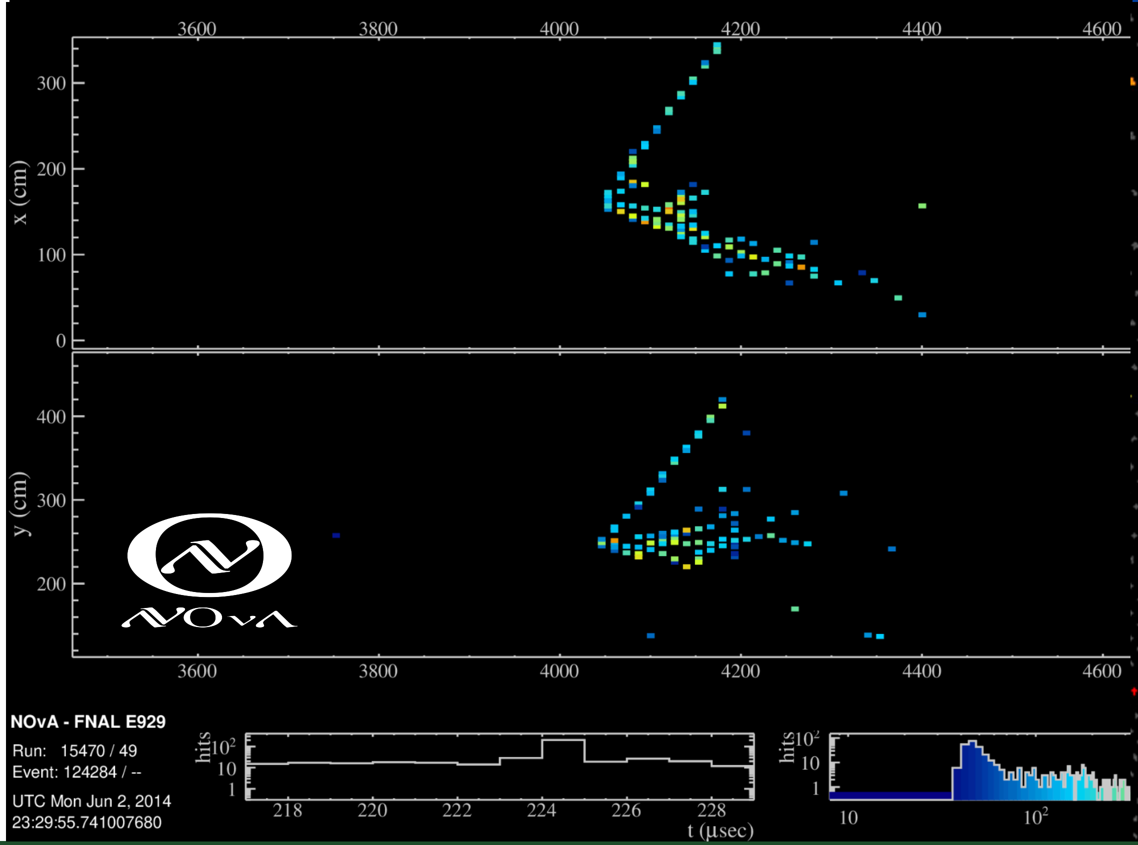
Fermilab SciDAC Scientific Discovery through Advanced Computing NOvA Colorado State University

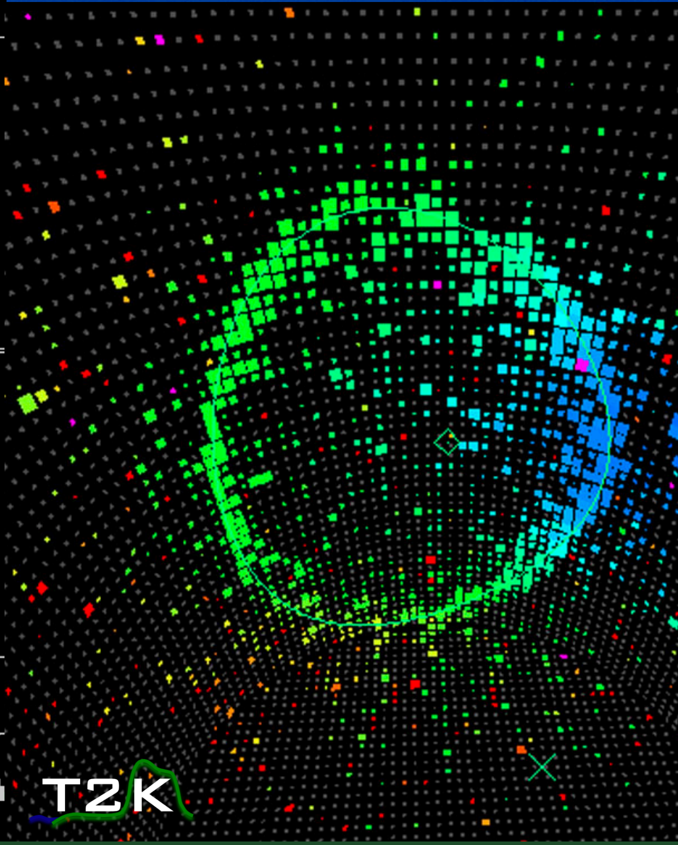
High Energy Physics Data

- High energy physics experiments collect massive amounts of data
  - Complicated event structure/ features
  - High intensity beams
- The first step of many analyses is often event selection and feature extraction
  - Highly-parallel task
  - Leverage high-speed interconnect and data storage at HPC facilities

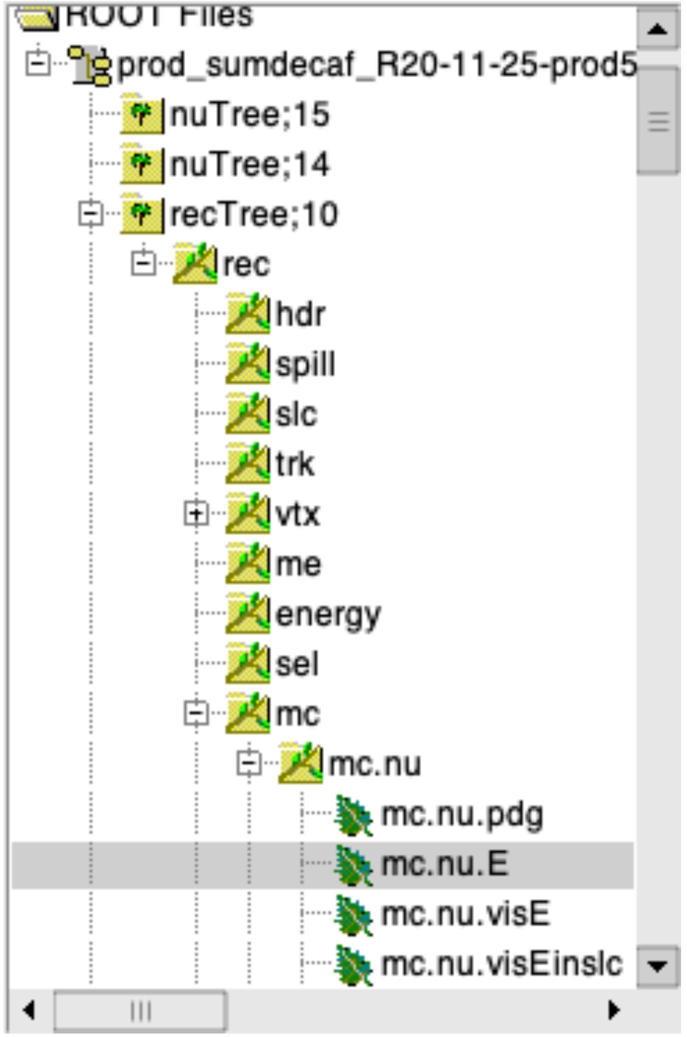








HDF5 Data Organization



- ROOT Files
  - prod_sumdecaf_R20-11-25-prod5
    - nuTree;15
    - nuTree;14
    - recTree;10
      - rec
        - hdr
        - spill
        - slc
        - trk
        - vtx
        - me
        - energy
        - sel
        - mc
          - mc.nu
            - mc.nu.pdg
            - mc.nu.E
            - mc.nu.visE
            - mc.nu.visEinslc

- HDF5 is the foundation of our Python analysis ecosystem
  - Can leverage highly optimized MPI implementations on HPC platforms
  - Large industry investment
- Data are store column-wise for efficiency columnar processing
- Flat tables, event indices associate features across tables
- From ROOT…
  - Nested tuples are flattened into separate groups (HDF5 Group)
  - Leaves compose the table's columns (HDF5 Dataset)

/rec.mc.nu (HDF5 Group)

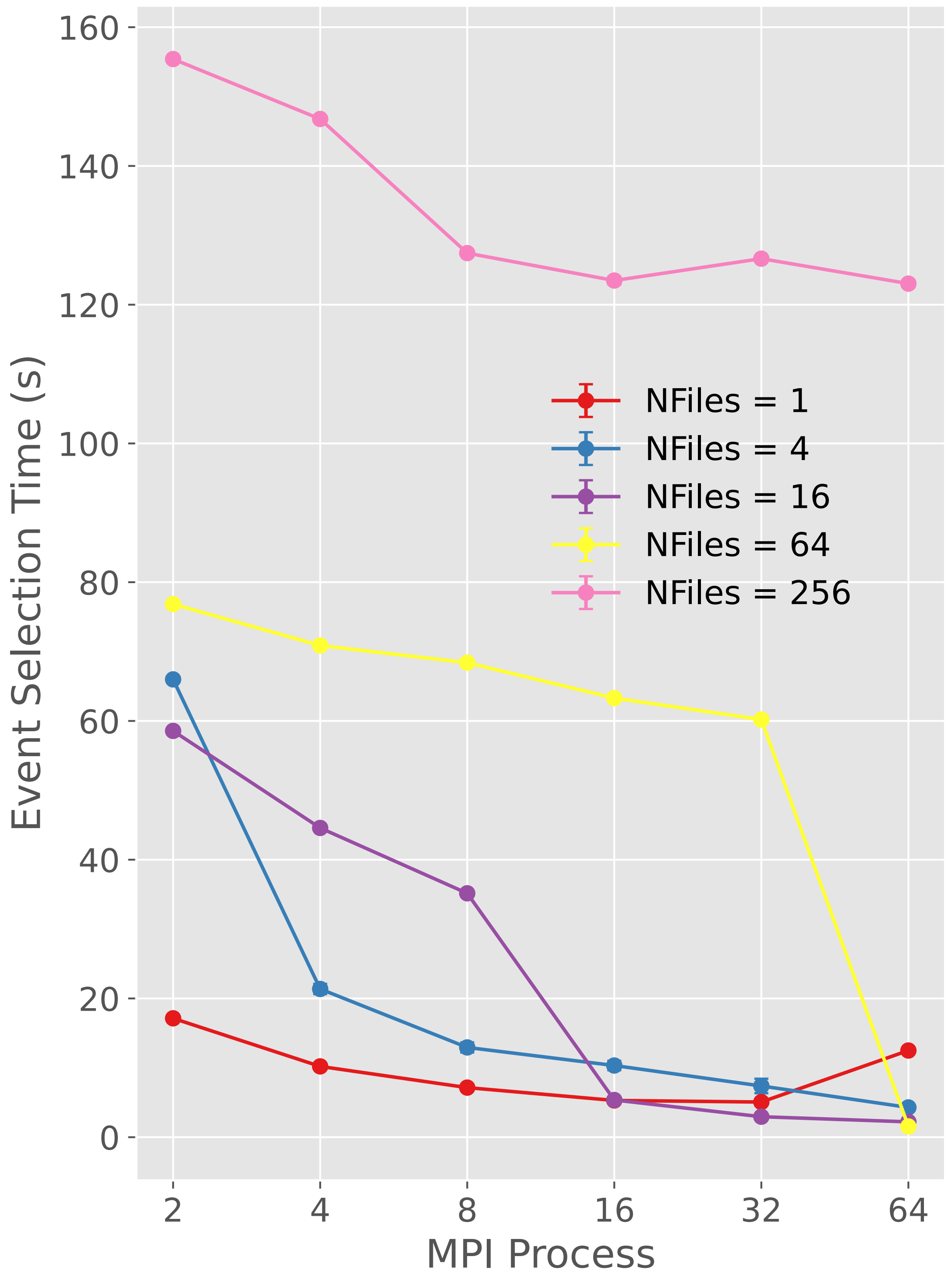
Event Index	pdg	E	visE	visEinslc	…
0	-14	1.6	0.89	0.79	
1	-14	2.3	0.92	0.91	
2	-14	2.07	1.26	1.23	
3	-14	2.00	1.04	0.98	

HDF5 Concatenation

https://github.com/NU-CUCIS/ph5concat

- HDF5 Chunking allows us to scale file size with little overhead
- Demonstrate scaling compression and improved event-selection time at various levels of concatenation of NOvA Monte Carlo dataset

- Monolithic dataset provides most flexibility
  - Consistent x10 speedup in event selection time
  - 11x compression
  - Self-contained
- Concatenated almost 3TB of uncompressed data using only 985 CPU hours (NERSC Cori)
  - $\mathcal{O}(10^2)$ GB file size enables laptop analyses

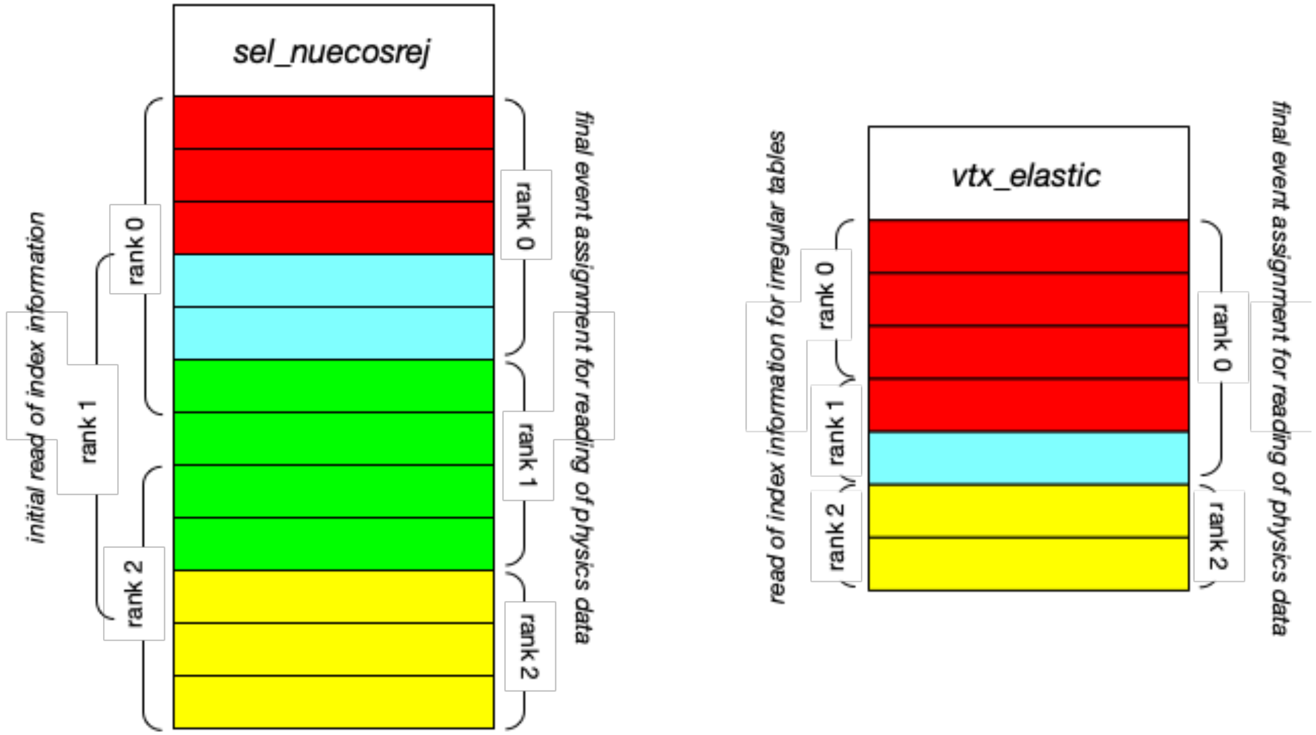


PandAna

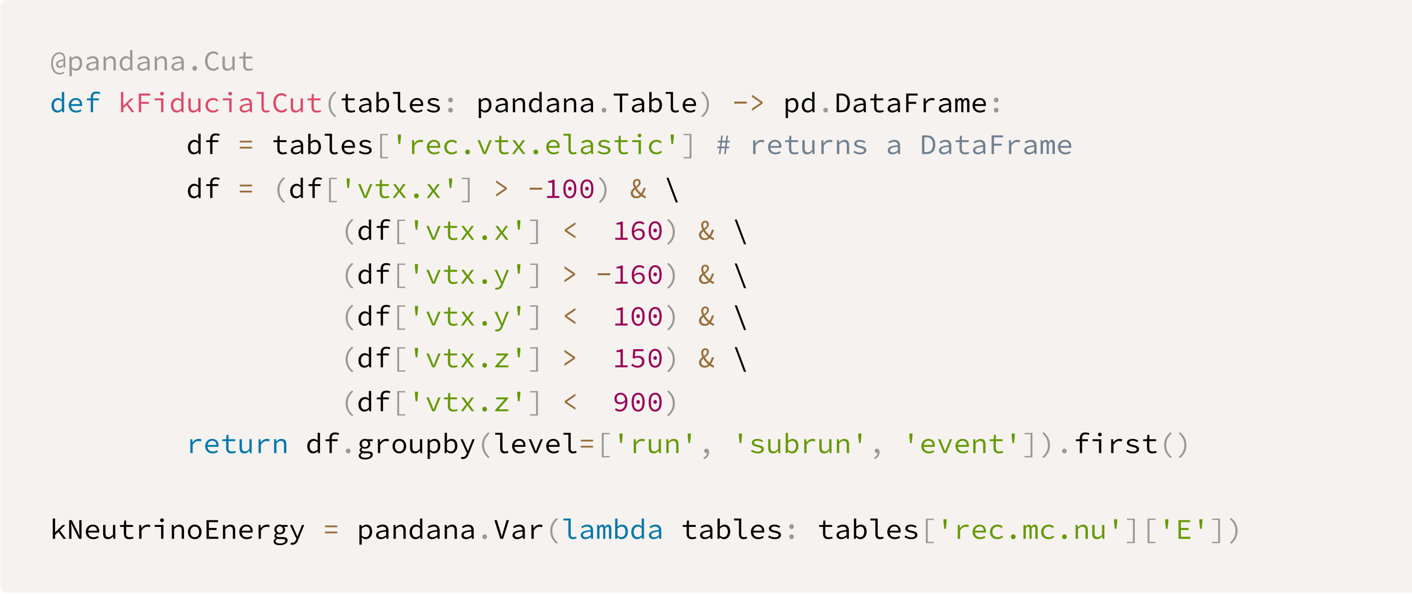
https://github.com/HEPonHPC/pandana

- PandAna is an implicitly parallel event selection framework
- Event indexing is used to distribute event selection workload among MPI ranks
  - Leverages parallel HDF5 I/O
  - Scalability limited by number of events
  - Handles irregular group sizes
- Easy to write, Pandas operations

- Possible improvements
  - Reduce event index redundancy requirements
  - Optimized data-flow using abstract syntax trees


sel_nuecosrej
initial read of index information
rank 0
rank 1
rank 2
rank 0
rank 1
rank 2
final event assignment for reading of physics data
read of index information for irregular tables
rank 0
rank 1
rank 2
vtx_elastic
rank 0
rank 2
final event assignment for reading of physics data

User code for defining data-filtering operation (Cut)



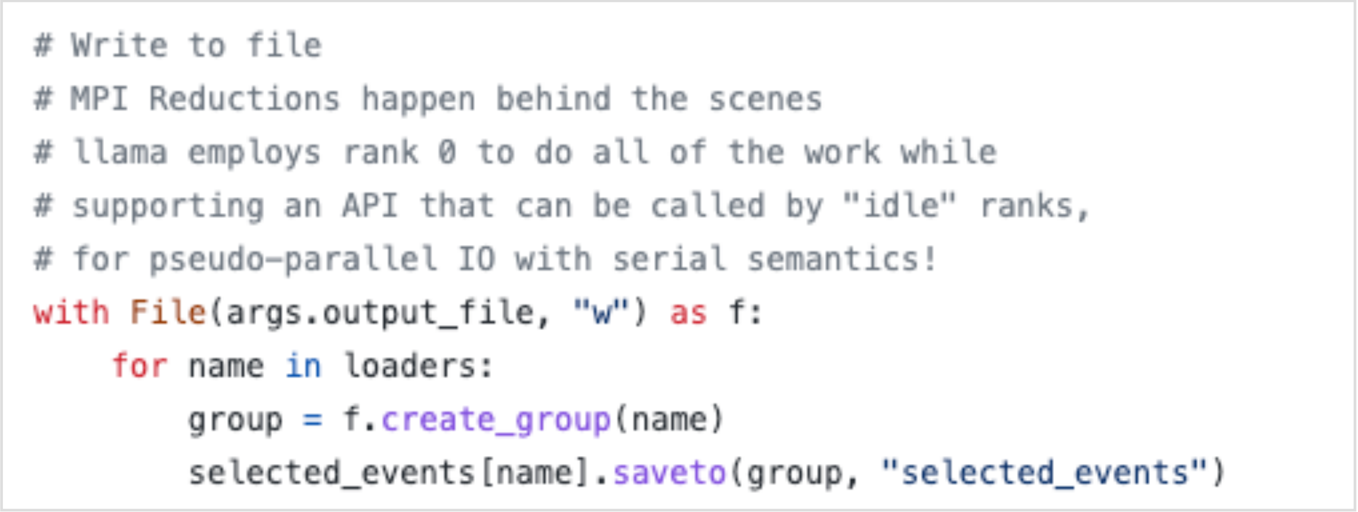
```
@pandana.Cut
def kFiducialCut(tables: pandana.Table) -> pd.DataFrame:
        df = tables['rec.vtx.elastic'] # returns a DataFrame
        df = (df['vtx.x'] > -100) & \
                 (df['vtx.x'] <  160) & \
                 (df['vtx.y'] > -160) & \
                 (df['vtx.y'] <  100) & \
                 (df['vtx.z'] >  150) & \
                 (df['vtx.z'] <  900)
        return df.groupby(level=['run', 'subrun', 'event']).first()

kNeutrinoEnergy = pandana.Var(lambda tables: tables['rec.mc.nu']['E'])
```

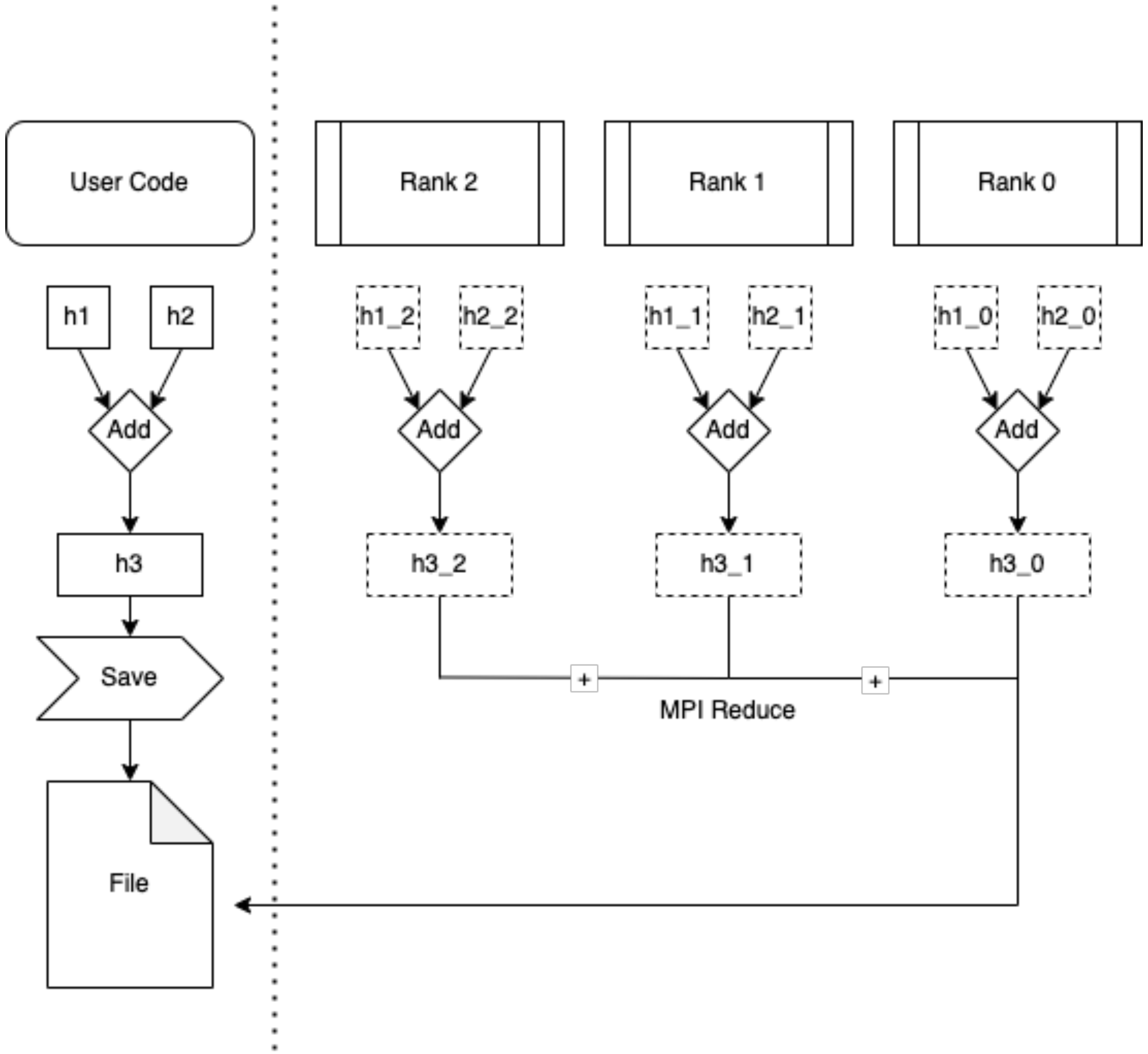
Llama — Llama does Lazy Mpi Aggregation

https://github.com/HEPonHPC/llama

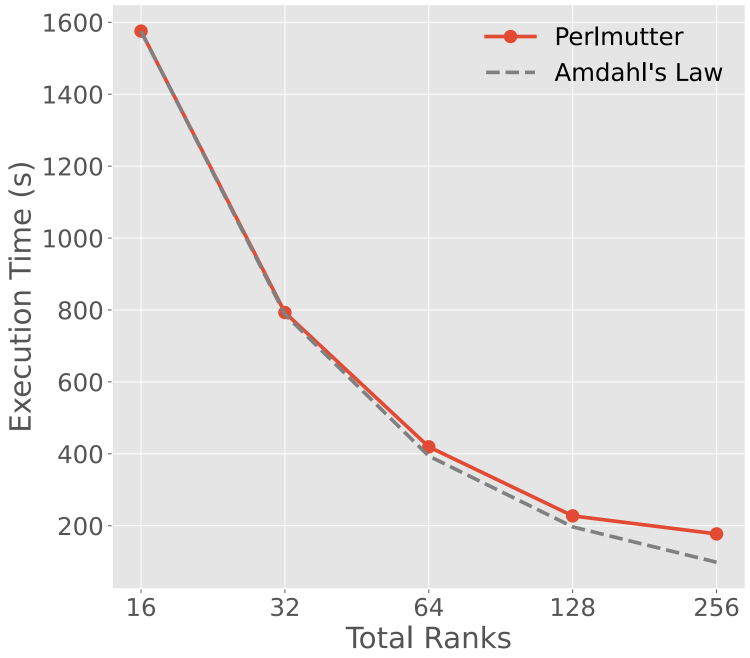
- Implicitly parallel histogram package
- Wraps boost_histogram for performance and uncertainty accumulation
- Lazy aggregation of common histogram operations
- Implements the Unified Histogram Interface for plotting



```
# Write to file
# MPI Reductions happen behind the scenes
# llama employs rank 0 to do all of the work while
# supporting an API that can be called by "idle" ranks,
# for pseudo-parallel IO with serial semantics!
with File(args.output_file, "w") as f:
    for name in loaders:
        group = f.create_group(name)
        selected_events[name].saveto(group, "selected_events")
```



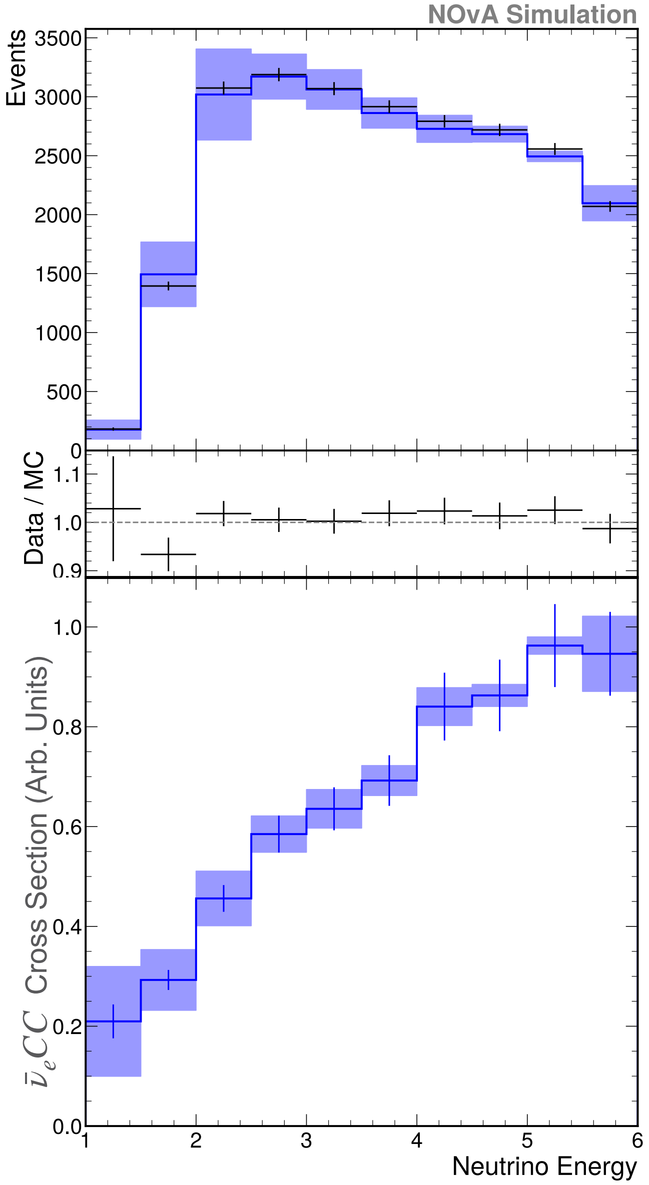
Case Study: A Realistic Analysis



Datasets	Beam Spills (million)	Size (GB)
Dataset 0	104	432
Dataset 1	25	74
Dataset 2	25	74
Dataset 3	25	76

*We demonstrate **near-perfect scalability** of our implicitly-parallel analysis workflow on a realistic sample size*

- NERSC's Perlmutter system
- $\bar{\nu}_e CC$ Inclusive cross-section measurement in the NOvA near detector
  - ~10,000 signal events
  - 12e20 Protons on Target (integrated luminosity) of data per Dataset
- Number of unique H5groups accessed = 16
- Number of unique H5datasets read = 54



Thank you

Derek Doyle	*derek.doyle@colostate.edu*
Micah Groh	*micah.groh@colostate.edu*
Jim Kowalkowski	*jbk@fnal.gov*
Andrew Norman	*anorman@fnal.gov*
Marc Paterno	*paterno@fnal.gov*
Saba Sehrish	*ssehrish@fnal.gov*