FERMILAB-SLIDES-21-112-SCD

The Story of HDF5 in High Energy Physics

Jim Kowalkowski, Marc Paterno and **Saba Sehrish**

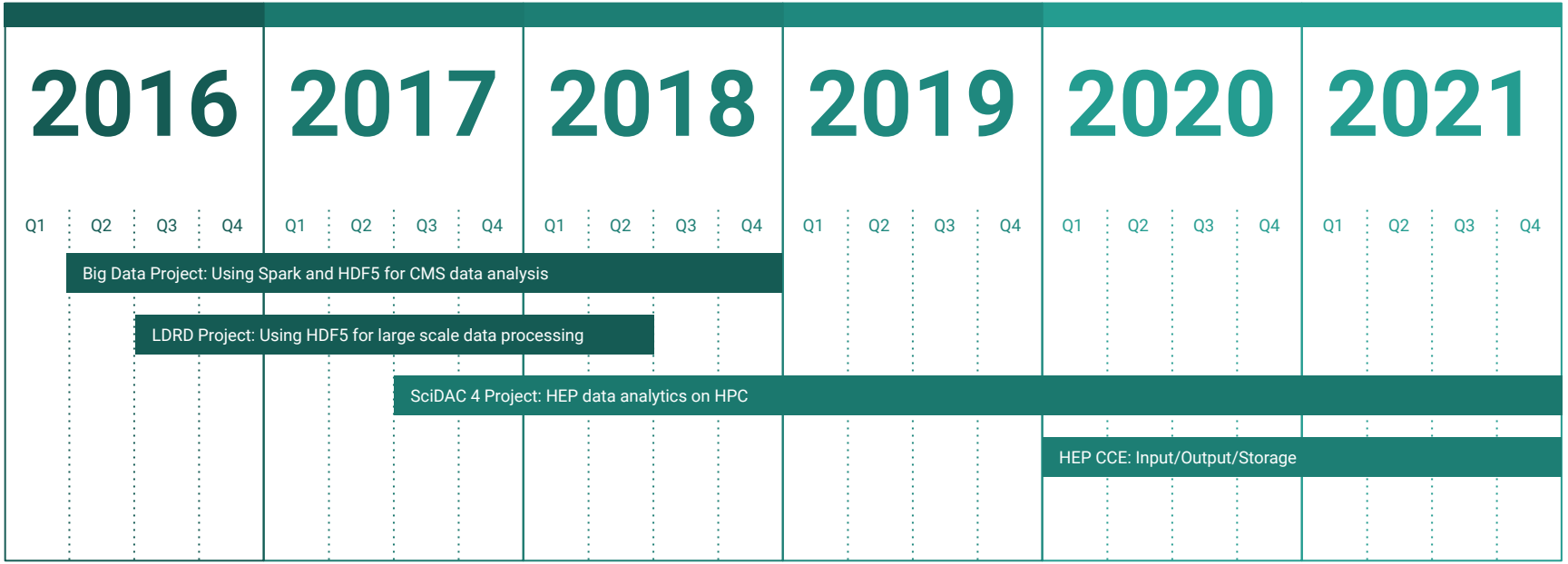
HDF User Group Meeting 2021

10/11/2021

Introduction

- For several years now, Fermilab has been investigating the use of HDF5 for large-scale analysis of experimental high energy physics (HEP) data.
- We had a variety of requirements for running jobs at HPC facilities as an alternative to grid-based processing; these were matched well by HDF5:
  - parallel writing capabilities,
  - efficient management and access to columnar data, and
  - compressed storage.
- We have now evaluated HDF5 in a wide range of HEP use-cases from raw detector data storage and retrieval to high-speed event selection during the later data analysis stages.
- Our goal has been to bring HDF5 into HEP as a standard tool for data storage and access.
- In this talk, we will present a historical view (or prehistoric view, or perhaps hysterical view), of work that has transpired, the current state of projects, and indicate what we see as useful future directions.

HDF5 projects timeline

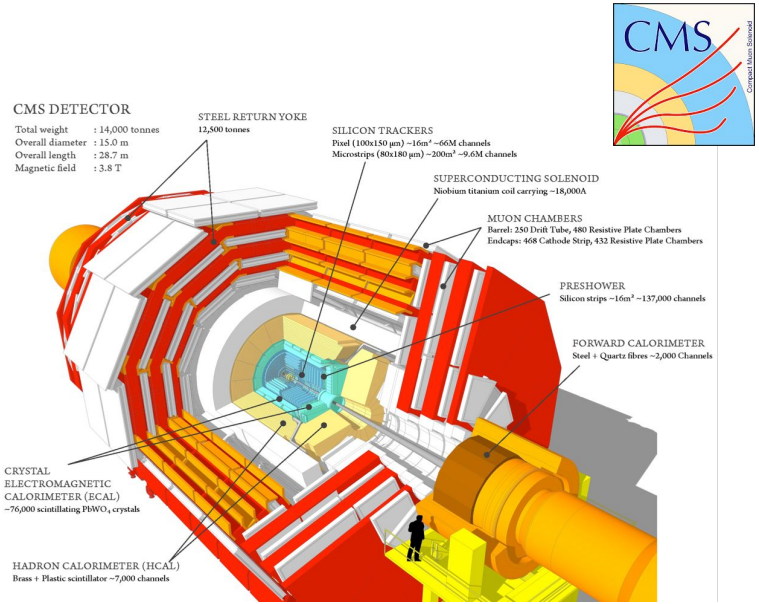


Fermilab

CMS Big Data Project: Spark and HDF5 for analysis

Can "Big Data" tools (e.g. Spark) and HPC resources benefit HEP's data- and compute-intensive statistical analysis to improve time-to-physics?

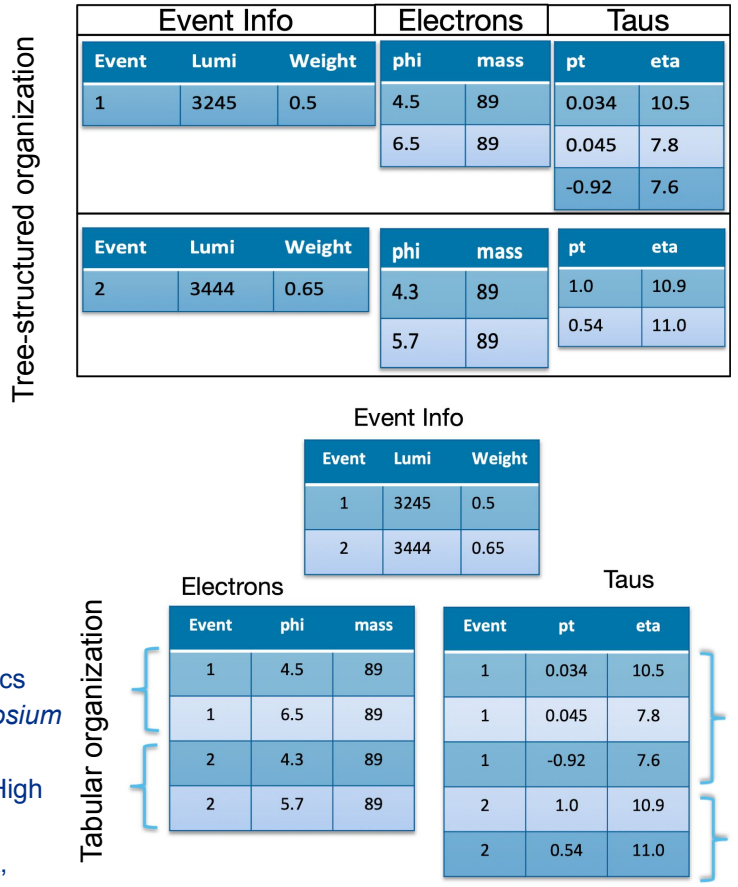
- Compact Muon Solenoid (CMS) is one of the major experiments at the Large Hadron Collider (LHC) at CERN. In 2012, CMS was co-discoverer of the Higgs boson.
- We partnered with CMS to investigate Spark as a parallel analysis tool for columnar data stored in HDF5.



Fermilab

Spark and HDF5

- HEP had traditionally used tree-structured data.
- Data organized in form of tables (HDF5 groups and datasets).
- Spark was not able to handle data stored in HDF5 format well. Reading and building dataframes was slow.
- MPI data parallel worked out extremely well with this data organization and resulting code was easy to follow.



Publications:

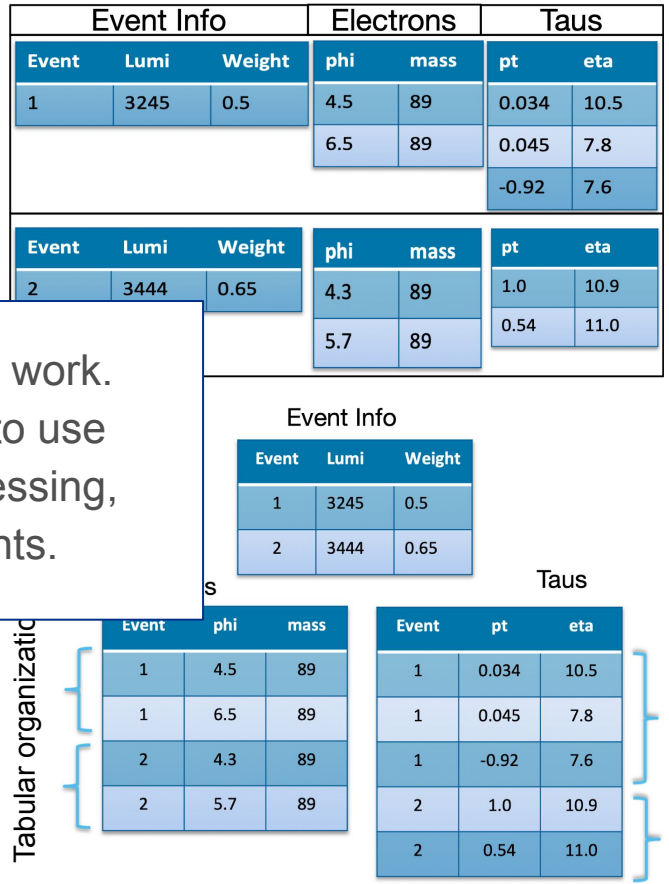
- S. Sehrish, J. Kowalkowski and M. Paterno, "Spark and HPC for High Energy Physics Data Analyses," *2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, Lake Buena Vista, FL, 2017, pp.
- S. Sehrish, J. Kowalkowski, M. Paterno, and C. Green. 2017. Python and HPC for High Energy Physics Data Analyses. In Proceedings of the 7th Workshop on Python for High-Performance and Scientific Computing (PyHPC'17). ACM, New York, NY, USA, Article 8, 8 pages 1048-1057.

Fermilab

Spark and HDF5

- HEP had traditionally used tree-structured data.
- Data organized in form of tables (HDF5 groups and datasets).
- Spark was not a
  HDF5 format we
  dataframes was
- MPI data paralle
  with this data o
  was easy to foll

Take away: HDF5 is the good part of this work. The findings here led to the LDRD work to use MPI and HDF5 for large scale data processing, which used data from different experiments.

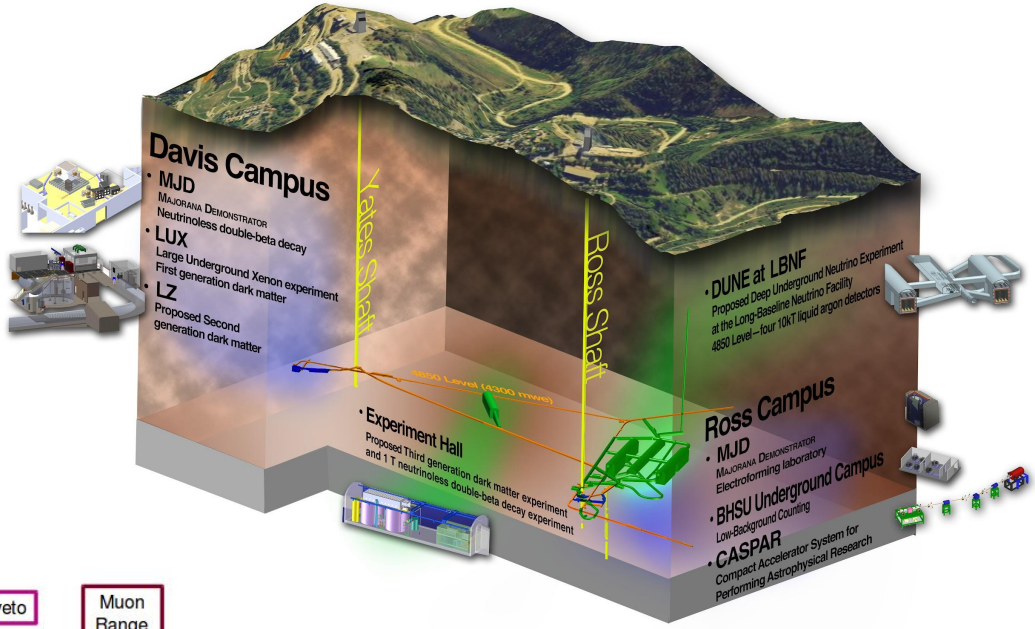


Publications:

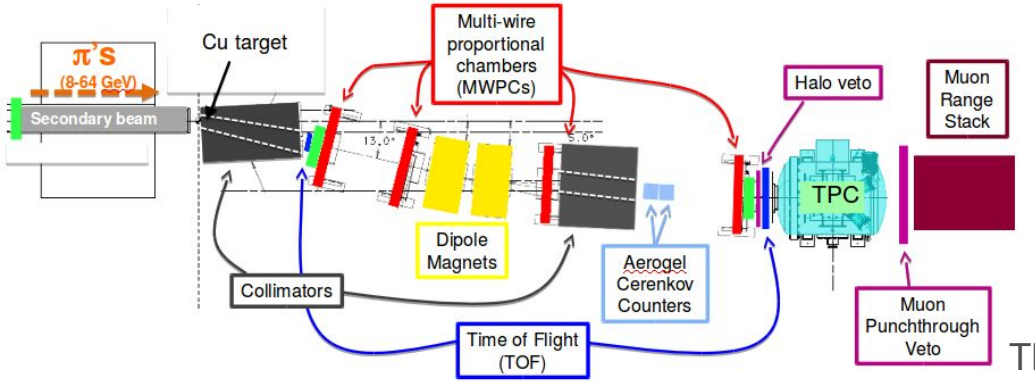
- S. Sehrish, J. Kowalkowski and M. Paterno, "Spark and HPC for High Energy Physics Data Analyses," *2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, Lake Buena Vista, FL, 2017, pp.
- S. Sehrish, J. Kowalkowski, M. Paterno, and C. Green. 2017. Python and HPC for High Energy Physics Data Analyses. In Proceedings of the 7th Workshop on Python for High-Performance and Scientific Computing (PyHPC'17). ACM, New York, NY, USA, Article 8, 8 pages 1048-1057.

Neutrino experiments

The main thrust of Fermilab today is neutrino physics. Experiments have run from small-scale test of detector technology to the largest neutrino experiments in the world. Neutrinos are the least-well understood particles in nature.



The DUNE detector in Lead, SD



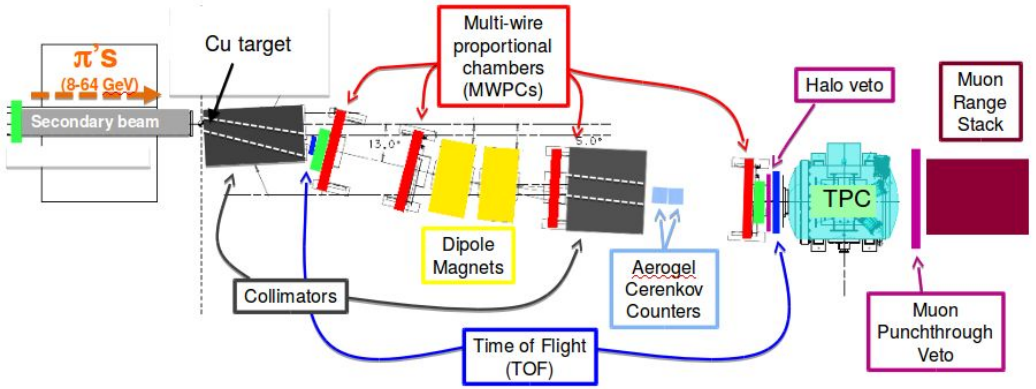
The LArIAT experiment at Fermilab

Fermilab

LDRD: Using LArIAT data, demo parallel processing

Can the Python ecosystem and HPC resources replace the traditional batch-oriented file processing used for large-scale HEP data processing?

- The Liquid Argon in a Testbeam (LArIAT) experiment's aim is to study particle tracks to better understand how different types of particles — in particular electrons and photons — interact in liquid argon, and how these interactions appear in the collected data. This will be used to guide future neutrino detector design.
- We partnered with LArIAT to evaluate the performance of data-parallel Python programs at scale.



Fermilab

High-performance use for columnar storage

- LArIAT data: 42 TB of digitized waveforms
  - 3072 samples per wire
  - 480 wires per event
  - More than 15 million events
- We developed a C++ library to make simpler the writing of tabular data into HDF5
  - We process the data one event at a time.
  - We want the file to contain datasets that span *all* the events.
  - This library is now used by several experiments (to be named later)
- We wrote the data to a single compressed HDF5 file: 4.2 TB on disk.
  - This was done before parallel writing of compressed data was working; it took 36 days of running in a single process to finish.

Parallel reading performance at scale

- Our demo program did the first few steps of LArIAT data processing:
  - read and decompress data
  - on each wire: use FFT, filtering, and inverse FFT to perform noise reduction
- All operations done in Python, using *mpi4py*, *h5py* and *numpy*.
- Run on KNL nodes on Cori, showed near perfect scaling from 200 to 1200 nodes.

Processes per node	Nodes	Processing time (s)	Normalized time
64	200	1068.0	1.000
64	1200	180.5	0.169

Publications:

- S. Sehrish, J. Kowalkowski, M. Paterno, and C. Green. 2018. Data Parallel Python for High Energy Physics Data Analyses. In Proceedings of the 8th Workshop on Python for High-Performance and Scientific Computing (PyHPC'18). ACM, New York, NY, USA.

Parallel reading performance at scale

- Our demo program did the first few steps of LArIAT data processing:
  - read and decompress data
  - on each wire: use FFT, filtering, and inverse FFT to perform noise reduction
- All operation … *numpy*.
- Run on KNL … ng from 200 to 1200 nodes.

Takeaway: Excellent reading performance for properly structured data in HDF5. The tabular organization in memory resulted in simple analysis code with implicit data parallelism and no MPI calls in physicists' code, as desired.

Processes per …	…	…	Normalized time
64	200	1068.0	1.000
64	1200	180.5	0.169

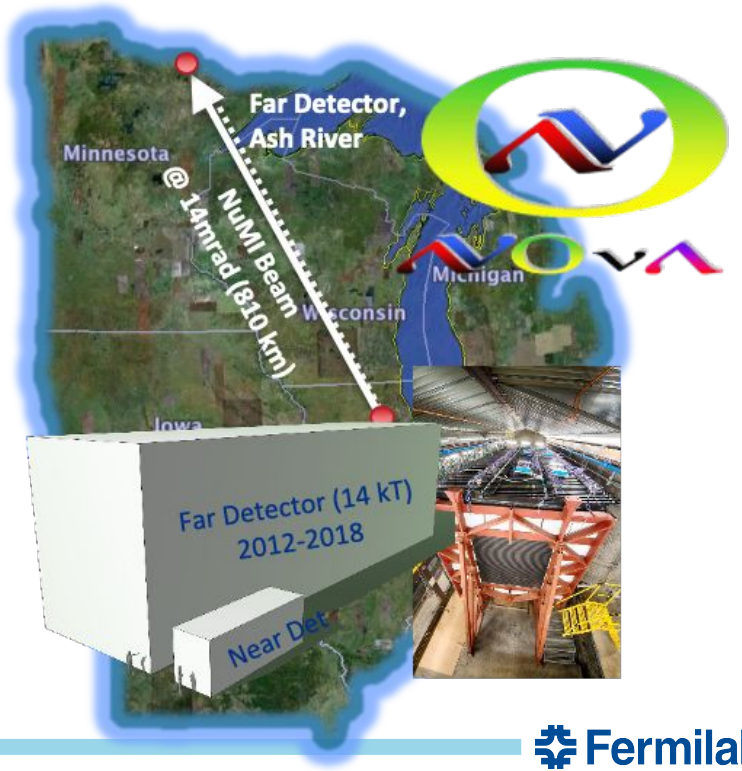
Publications:

- S. Sehrish, J. Kowalkowski, M. Paterno, and C. Green. 2018. Data Parallel Python for High Energy Physics Data Analyses. In Proceedings of the 8th Workshop on Python for High-Performance and Scientific Computing (PyHPC'18). ACM, New York, NY, USA.

A user-friendly analysis environment (NOvA)

Goal: Support development of a user-friendly analysis environment using Python data science tools and columnar data stored in HDF5.

- The NuMI Off-axis ν_e Appearance (NOvA) experiment is making precise measurements describing how muon neutrinos created at Fermilab transform into electron neutrinos detected at the site 810 km away.
- The first adopters of HDF5 “ntuples” (tabular data) were NOvA experimenters: they created the PandAna framework.



Fermilab

Using the tabular organization of data for Analysis

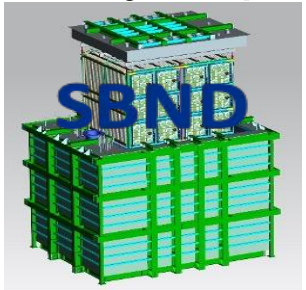
- We have provided support for storing NOvA's Common Analysis Format (CAF) equivalent data in HDF5
- NOvA has been writing HDF5 analysis ntuples in production since 2018
  - NOvA collaborators have been this in analysis for even longer.
- PandAna meets our ease-of-use goal
  - high-level manipulations; "whole array" manipulations rather than loop; *numpy* and *pandas*.
  - Python viewed as "friendlier" than C++: easier for new grad students to learn.
- Serial performance was surprisingly good:
  - Users report a faster development cycle compared to C++ (no compilation, interactive exploration)
  - Users report 5-100 times *faster* than traditional compiled C++ code for some analyses

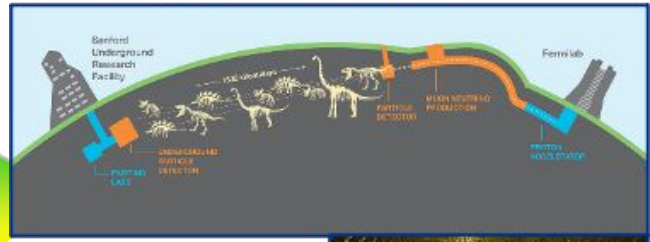
Using the tabular organization of data for Analysis

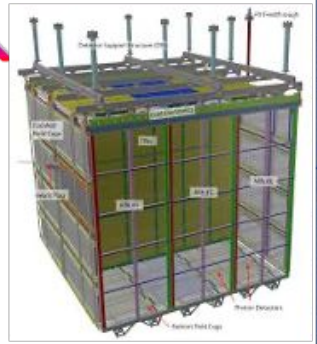
- We have provided support for storing NOvA's Common Analysis Format (CAF) equivalent data in HDF5
- NOvA has been writing HDF5 analysis ntuples in production since 2018
  - NOvA collabora onger.
- PandAna meets ou
  - high-level manip ther than loop; *numpy* and *pandas*.
  - Python viewed as "friendlier" than C++: easier for new grad students to learn.
- Serial performance was surprisingly good:
  - Users report a faster development cycle compared to C++ (no compilation, interactive exploration)
  - Users report 5-100 times *faster* than traditional compiled C++ code for some analyses

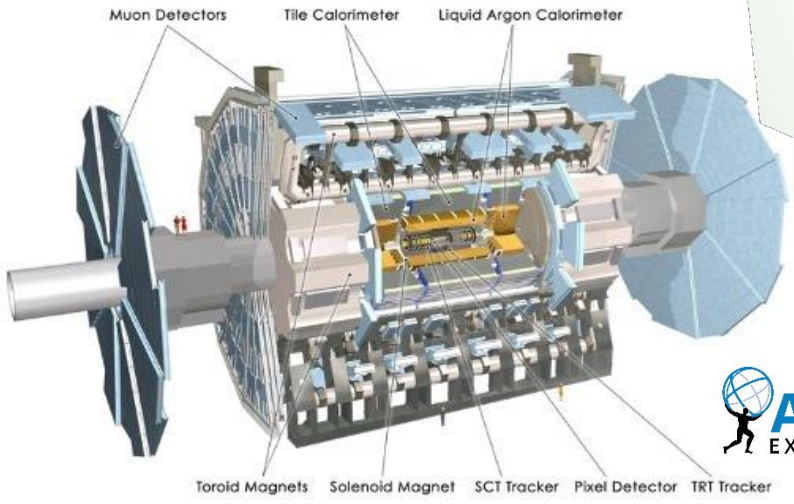
Takeaway: PandAna style of writing analysis code is popular and user friendly. Serial performance is good. Note that it was designed to allow data-parallel processing.

Our SciDAC project is working with many experiments









ATLAS EXPERIMENT

SciDAC HEP on HPC

- The goal of the SciDAC-4 *HEP Data Analytics on HPC* is to extend the physics reach of LHC and neutrino physics experiments by collaborating with ASCR scientists and utilizing the computing power and scale of national HPC facilities.
- We have made significant steps to effectively use state-of-the-art HPC centers to solve HEP problems by better integrating analysis workflows onto HPC systems, and blending HPC software and services directly into the HEP applications.
  - Use “native” HPC tools and techniques to solve HEP problems.
  - Utilize HDF5 as a storage format and tool for parallel access to analysis data.

Collaboration with Northwestern University (NU) on parallel writes

Goal: Develop scalable parallel I/O utility programs to concatenate large volume of HEP experimental data on DOE leadership HPC computers

- The nature and context and constraints of the earlier steps of the analysis is such that it is not going to be run at HPC centers, and files will be too small in size and large in number for parallel analysis.
- We want small number of large files for what we are looking to do.
- We need this fast scalable concatenation to make these files.
- The research details on this work were covered by Sunwoo Lee's talk this morning.

Collaboration with Northwestern University (NU) on parallel writes

Goal: Develop scalable parallel I/O utility programs to concatenate large volume of HEP experimental data on DOE leadership HPC computers

- The nature and ps of the analysis is such that it is les will be too small in size and larg
- We want smal king to do.
- We need this f files.
- The research details on this work were covered by Sunwoo Lee's talk this morning.

- Concatenation of NOvA experimental data enables high degree of computational parallelism *in analysis*.
- Parallel data concatenation achieved scalable performance on Cori.

PandAna 2

- In our SciDAC project, we have developed an easy-to-use environment for fast and scalable analysis.
  - easy-to-use: Python, and Python data science tools (e.g. numpy, pandas);
  - fast: natively data-parallel and taking advantage of HPC features;
  - scalable: same code works on laptops, clusters, HPC systems.
- Our plan is for a tool that is *not* specific to any experiment, nor to just one HEP community (collider or neutrino).
- We enhanced PandAna to use MPI for data-parallel processing
  - Only PandAna2 library code uses MPI, for parallel reading and data distribution
- End-user code is implicitly parallel, and does not use any MPI calls

PandAna 2

- In our SciDAC project, we have developed an easy-to-use environment for fast and scalable analysis
  - easy-to-use: e.g. numpy, pandas);
  - fast: natively PC features;
  - scalable: sam systems.
- Our plan is for a ment, nor to just one HEP community (colli
- We enhanced Pa rocessing
  - Only PandAn ding and data distribution
- End-user code is implicitly parallel, and does not use any MPI calls

Takeaway: In-memory data processing code scales perfectly. We are struggling to make reading scalable. Working in collaboration with NU to address parallel IO performance issues.

Parallel reading in PandAna 2

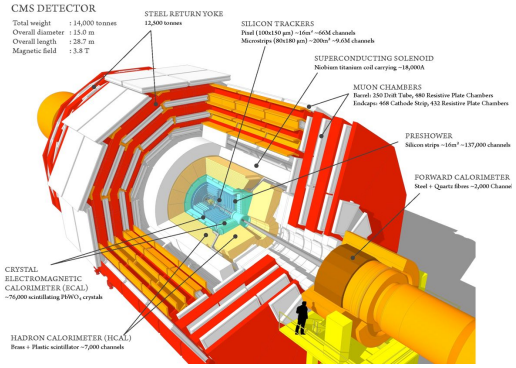
- Distribute data equally among all ranks so no communication is necessary
- In working with NU group, we discovered that this pattern of reading interacted poorly with compressed therefore chunked data sets.
- Using as many MPI ranks as we would like to use for processing resulted in reads that were too small and not give any performance scalability.
- We were able to do some tuning of the chunk sizes to improve the situation but this is not enough
- We are working on a revision of PandAna, and working on its design to address these issues in multiple ways
  - Reduce number of reads
  - Make individual reads larger

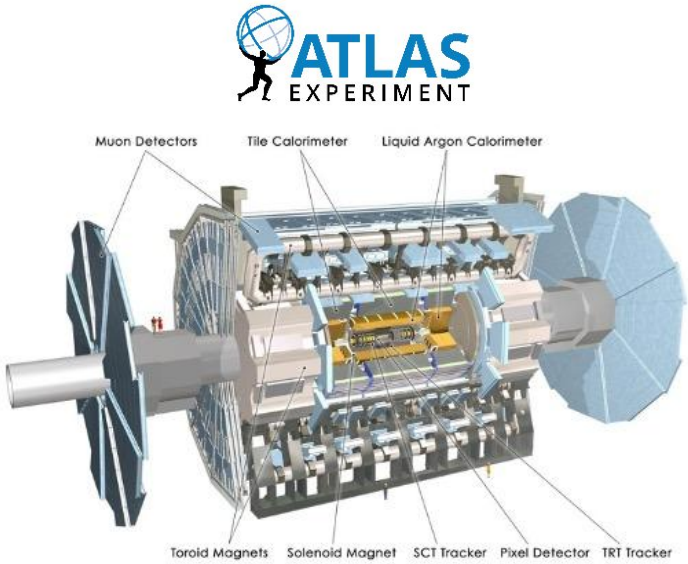
The Exa.TrkX ML project and HDF5

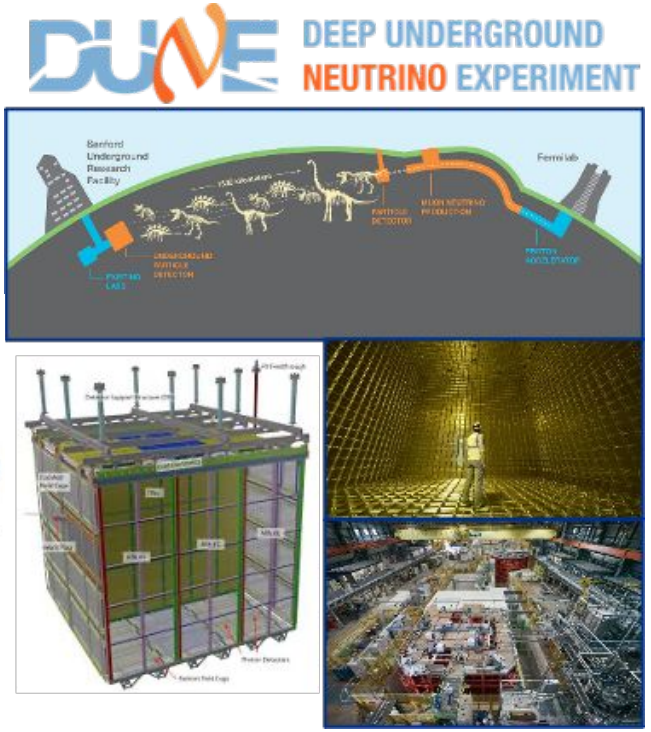
- The goals of Exa.TrkX are to develop production-quality Machine Learning models for charged particle tracking, and to utilize distributed training with graph neural networks (GNNs) on HPC facilities.
  - applying models to track and shower reconstruction for DUNE.
- University of Cincinnati developing numl, a package for producing HDF5 files for ML directly from experiment data processing frameworks.
  - Contains full simulation truth information needed for training
  - Includes pynuml for support of GNN workflows
- Working with NU to develop efficient methods for reading collections of neutrino events necessary for training GNN models

HEP Center of Computational Excellence (CCE)

HEP-CCE







U.S. DEPARTMENT OF ENERGY | Office of Science

Argonne NATIONAL LABORATORY | BROOKHAVEN NATIONAL LABORATORY | Fermilab | BERKELEY LAB Bringing Science Solutions to the World

CCE IO Storage

- In our SciDAC project, we have demonstrated the efficient and high performing data access and hence subsequent analysis by using HDF5 representation of the *analysis-ready* data.
- In the IOS project under HEP CCE, by utilizing the established expertise, we are evaluating the use of HDF5 for intermediate data storage *unlike analysis-ready data in tabular form*.
- We are interested in developing an experiment-independent parallel HDF5 IO approach.
  - Write HEP “data products” (representing raw detector readouts, identified energy deposits, particle trajectories, *etc*.) that have already been serialized using ROOT to HDF5

IOS: Current Status

- Using data that is already serialized with ROOT facilities allowed us to come up with a general layout
- We are using two HDF5 data sets to represent one data product type in an event
  - One data set carrying actual serialized byte stream and the other data set corresponding offsets defining event boundaries
  - Allows to read data back either one data product type for all events Or all data products for one event
- We are currently using a mini test Framework that was developed as a part of CCE work. The multithreaded framework is designed to serialize events concurrently.
  - Supports different modes of input and output, such as ROOT, HDF5, etc.
  - Goal is to understand and compare performance differences among different IO modes
- We have used HighFive library in the first prototype, and later switched to C API with minimal C++ wrappers

IOS: Current status and next steps

- We have used data (reconstruction-level) from ATLAS, CMS and DUNE in our initial design and testing
- We are currently working on a parallel IO prototype to be able to simultaneously write events to HDF5 files
- We are able to run our serial HDF5 output/input mode on Cori
- Next:
  - Performance studies to establish baseline on Cori
  - Explore alternate layouts for writing to HDF5 datasets
  - Performance studies on parallel design

IOS: Current status and next steps

- We have used data (reconstruction-level) from ATLAS, CMS and DUNE in our initial design and testing
- We are curr[illegible]totype to be able to simultane[illegible]
- We are able[illegible]out mode on Cori
- Next:
  - Performance studies to establish baseline on Cori
  - Explore alternate layouts for writing to HDF5 datasets
  - Performance studies on parallel design

Takeaway: This use of HDF5 is integrated with ROOT, and is under evaluation. We will explore alternate approaches and HDF5 layout in collaboration with HDF5 team at LBNL.

Future directions we'd like to see for HDF5

- We need improved parallel-writing performance for compressed data
  - Addressed at this meeting by Sunwoo Lee
- We need efficient storage and reading of sparse data
  - Most NOvA datasets, for example, would be naturally addressed at 6D arrays, but they would be sparse arrays.
  - Efficient storage and reading of sparse arrays would allow us to simplify our code.

Future directions for our work

- We are exploring a different computational model for PandAna that would allow fewer and larger reads, to overcome the problem we face with reading compressed blocks.
- We are interested in evaluation of alternative C++ interfaces to HDF5 (e.g. h5cpp) to understand what advantages they have for our community.
- DUNE is interested in using the MPI-based parallel writing ability of HDF5 for event building.
- We have greatly benefitted from working with our colleagues at NU, and plan to continue doing so.

Conclusion

- HPC facilities are increasingly important to our community (HEP).
- We argue that doing HEP tasks using HPC-native technologies (as opposed to porting our community tools to run, somehow, on HPC) is the best way forward.
- We have found HDF5 to be a powerful library that provides good performance and solves important problems.
- Our use cases are stressing some aspects of HDF5 that have not, in the past, been so severely stressed. We would like to continue working with the HDF5 community to improve these aspects of the tools.

Acknowledgements

The work presented in this talk 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.
- The work presented in this talk is supported by the U.S. Department of Energy, Office of Science, Office of High Energy Physics, High Energy Physics Center for Computational Excellence (HEP-CCE) at Argonne National Laboratory, Fermi National Accelerator Laboratory, and Lawrence Berkeley National Laboratory.

This research used resources of

- The National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory.
- The Fermi National Accelerator Laboratory (Fermilab), a U.S. Department of Energy, Office of Science, HEP User Facility. Fermilab is managed by Fermi Research Alliance, LLC (FRA), acting under Contract No. DE-AC02-07CH11359, and includes funding from Fermilab LDRD program.