Scientific Computing Division

Lisa Goodenough

53rd Annual Users Meeting

11 August 2020

SCD: Who We Are and What We Do

We are Experimentalists and Theorists; Scientists, Postdocs, and Students; Particle Physicists, Computer Scientists, Engineers, Accelerator Physicists, Computational Physicists…

Fermilab

SCD: Who We Are and What We Do

We are Experimentalists and Theorists; Scientists, Postdocs, and Students; Particle Physicists, Computer Scientists, Engineers, Accelerator Physicists, Computational Physicists…

We support the mission of scientific discovery at Fermilab by providing advanced, innovative computing technology and services.

- **Accelerator Modeling**
- **Algorithms** for faster computing and better precision **(Reconstruction, Analysis)**
- Artificial Intelligence (AI): machine learning (Aleksandra Ciprijanovic's talk, next +1)
- DAQ
- Data storage and management
- Network resources
- Software tools and frameworks
- **Workflow management (HEPCloud)**
- Quantum Computing (Farah Fahim's talk next)

A Fast Changing Environment

- ➡ The **push to improve the physics** reach of our experiments is driving up the size of detectors and the intensity of particle beams in our next generation of experiments. These will give rise to unprecedented amounts of data to be processed, transferred, analyzed, stored…
- ➡ Computing technology is changing rapidly.

- **Technology changes we must keep up with:**
  - changes in software norms and conventions: e.g. Python has been gaining in popularity over the past 5 years
  - computing architectures are constantly changing: can require very different code and configurations to use effectively
  - how we package and build our software applications: e.g.containers now the norm
  - DOE is investing a lot of money in high performance computing (HPC) centers - NERSC, ALCF, OLCF - they are there for us to use!

The Future of HEP Computing: HPCs

High Performance Computing is fundamentally different from High Throughput Computing, the computing paradigm for HEP for the past two decades.

- **HTC involves executing a large number of loosely coupled tasks**
  - independent, sequential jobs
  - can be scheduled on many different computing resources across administrative boundaries
- **HPC involves executing tightly coupled, parallel jobs**
  - must run on a particular resource and need low-latency interconnects

Grid Computing Center

Theta at ALCF

The Future of HEP Computing: HPCs

DOE Leadership Computing Facilities: Argonne (ALCF) and Oakridge (OLCF)

ALCF: Theta (current)

- 11.60 Petaflops (10^{15} floating point ops/sec)
- 4392 nodes, each with one Intel KNL CPU (64 cores/node w/4x hardware threads)
- 192 GB DDR4 + 16 GB MCDRAM/node
- vectorization

OLCF: Summit (current)

- 193 Petaflops
- 4603 nodes, each with two IBM Power9 CPUs and 6 NVIDIA VOLTA GPUs
- 512 GB DDR4 + 96GB HBM2 RAM/node

ALCF: Aurora (2021)

- **>1 Exaflops (10^{18} floating point ops/sec) EXASCALE!**
- hybrid architecture, no vectorization
- nodes: two Intel Sapphire Rapids CPUs with six Intel XE GPUs

Fermilab

The Future of HEP Computing: HPCs

DOE Leadership Computing Facilities: Argonne (ALCF) and Oakridge (OLCF)

ALCF: Theta (current)

- 11.60 Petaflops (10^{15} floating point ops/sec)
- 4392 nodes, each with one Intel KNL CPU (64 cores/node w/4x hardware threads)
- 192 GB DDR4 + 16 GB MCDRAM/node
- vectorization

OLCF: Summit (current)

- 193 Petaflops (#2 in world)
- 4603 nodes, each with two IBM Power9 CPUs and 6 NVIDIA VOLTA GPUs
- 512 GB DDR4 + 96GB HBM2 RAM/node

ALCF: Aurora (2021)

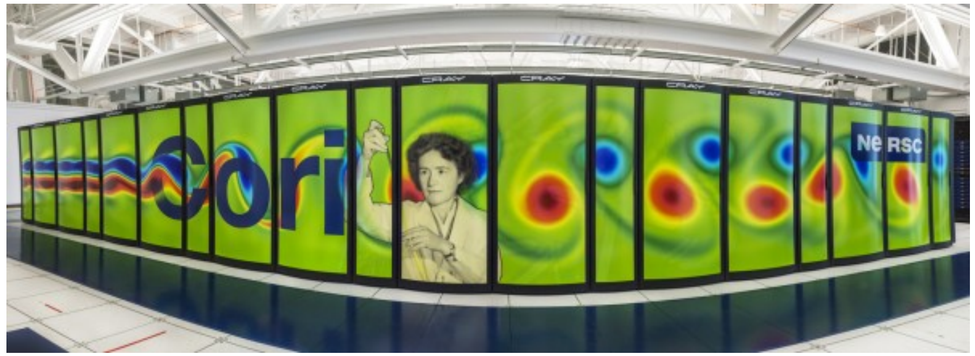
- **>1 Exaflops (10^{18} floating point ops/sec) EXASCALE!**
- hybrid CPU/GPU architecture, no vectorization
- nodes: two Intel Sapphire Rapids CPUs with six Intel XE GPUs

The Future of HEP Computing: HPCs

National Energy Research Scientific Computing Center: NERSC

NERSC: Cori (current)

- ~30 Petaflops
- 2388 Intel Haswell nodes (32 cores/node w/2x hyper-threading), 112 GB DDR4 + eight 16 GB DIMMS/node
- 9668 Intel KNL nodes (68 cores/node w/4x hardware threads), 96 GB DDR4 + six 16 GB DIMMS/node
- vectorization



NERSC: Perlmutter (2020-2021)

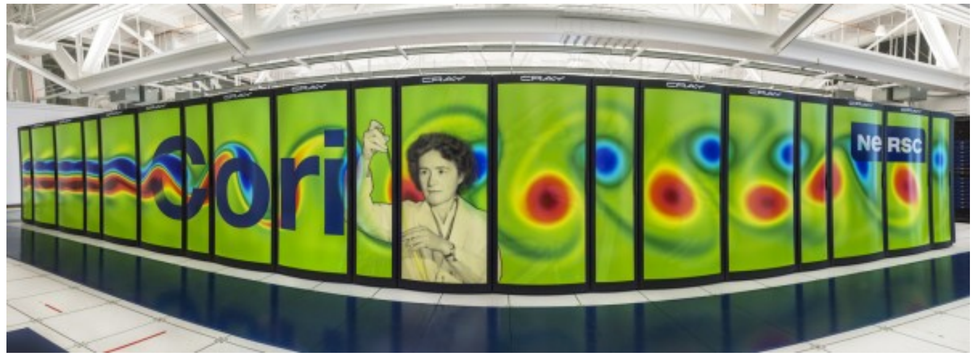
- ~100 Petaflops
- hybrid CPU/GPU architecture

The Future of HEP Computing: HPCs

National Energy Research Scientific Computing Center: NERSC

NERSC: Cori (current)

- ~30 Petaflops
- 2388 Intel Haswell nodes (32 cores/node w/2x hyper-threading), 112 GB DDR4 + eight 16 GB DIMMS/node
- 9668 Intel KNL nodes (68 cores/node w/4x hardware threads), 96 GB DDR4 + six 16 GB DIMMS/node
- vectorization



Trend is from multi-threading and vectorization on modern multicore CPUs to GPUs, or a heterogenous model using CPUs and GPUs.

NERSC: Perlmutter (2020-2021)

- ~100 Petaflops
- hybrid CPU/GPU architecture

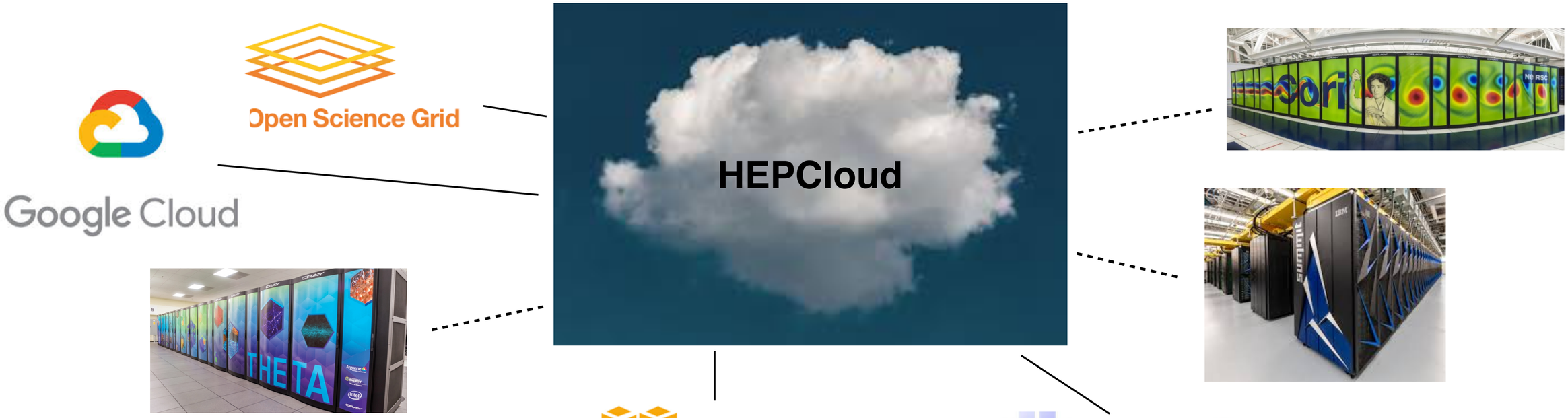
Our challenge is to keep up with the changes!

Fermilab

HEPCloud: a Unified Access Portal

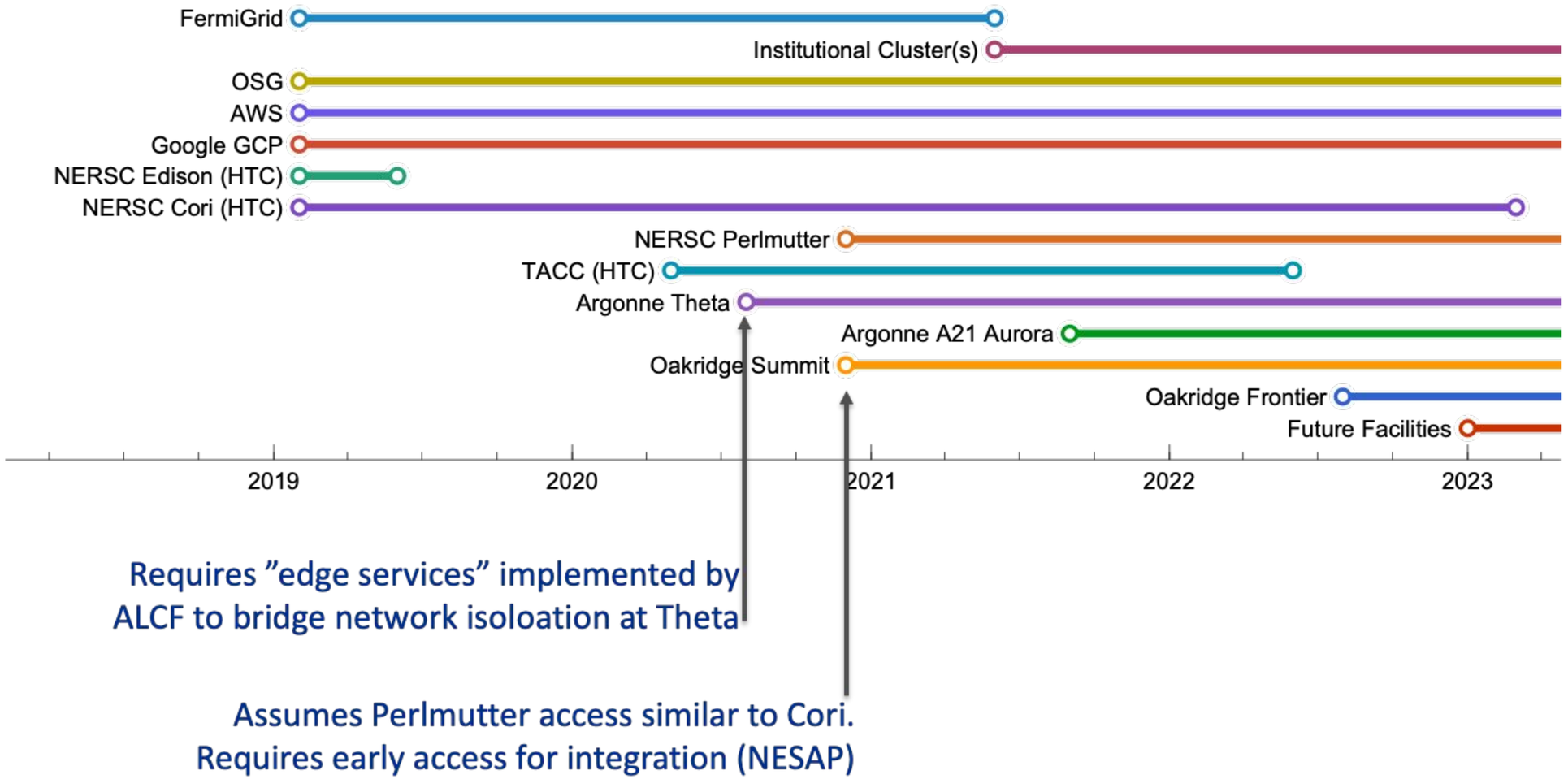
HEPCloud is a cloud computing service for HEP experiments at Fermilab.

It provides (or will provide) a **unified portal to access computing** on traditional Fermilab grid resources (FermiGrid, OSG), as well as novel resources such as commercial or academic clouds (Amazon, Google), and HPC resources at supercomputing centers (NERSC, ALCF, OLCF).



Fermilab

HEPCloud Site Integration Roadmap



from FIFE Roadmap 2020 talk:
https://indico.fnal.gov/event/43634/contributions/187643/attachments/130980/160001/FIFE_Roadmap_Summer_2020.pdf

HEPCloud: Users' Experience

- HEPCloud will provide **access to resources** from outside Fermilab (either 'rental', through collaboration, or through grant proposals) in a manner that is cost effective and **transparent to the user**.
- HEPCloud will provide **resources matching requirements** to workflows (*e.g.* memory, architecture (GPU?, CPU?), available allocations, funding, storage…). Resources will include:
  - High Throughput Computing (HTC) resources
  - multi-node resource blocks for the purpose of running MPI style jobs
  - resource blocks for "pipeline" workflows in which individual components of a pipeline are setup and communicate via the filesystem.
  - HEPCloud will direct all other workflows to resources as determined by HEPCloud.
- Experiment directed and **managed workflows will have the option to specify where they will run**.

HEP Event Reconstruction with Cutting Edge Computing Architectures (a SciDAC Project)

Goal: to increase the utilization of parallel computing architectures in HEP **reconstruction**, particularly for CMS and neutrino experiments using Liquid Argon Time-Projection Chamber (LArTPC) detectors

- **Key algorithms in reconstruction workflows have been identified and redesigned**
  1. Charged particle track reconstruction for CMS
  2. Hit finding for LArTPC detectors such as ICARUS and MicroBooNE
- **Modified algorithms are a factor of 6-12x faster than the original algorithms on a single thread** (additional speedups occur when running on multiple threads)
- Portable implementations of the algorithms for use at supercomputers and with heterogenous platforms have been explored

See https://indico.cern.ch/event/868940/contributions/3814728/ for more details

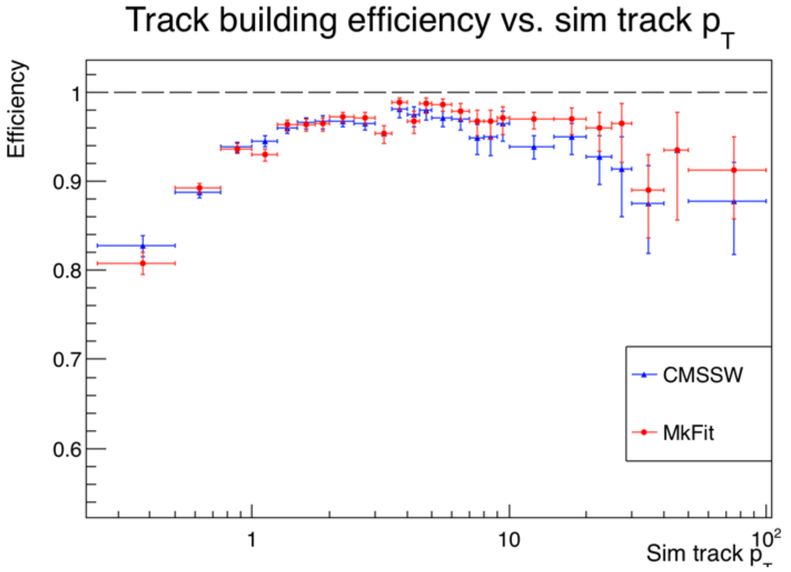
Fermilab

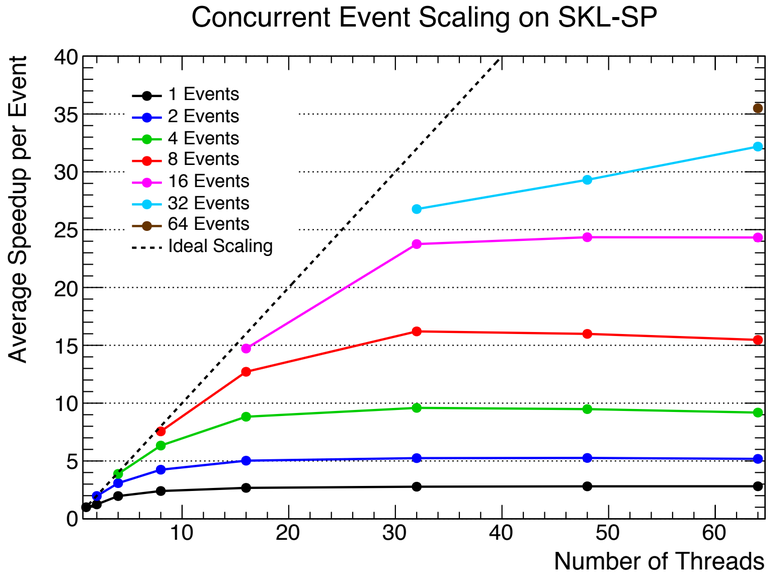
thanks to Giuseppe Cerati for slide contents

HEP Event Reconstruction: 1. Charged Particle Track Reconstruction for CMS (mkFit)

Tracking is the dominant contributor to the reconstruction time. But track building is a combinatorial algorithm that has traditionally been implemented in serial fashion - challenging to parallelize.

- mkFit track building **achieves comparable physics performance** as standard CMS tracking (Icarus on the horizon)
- Standalone application achieves **speedups of up to 3x from vectorization and up to 35x from multithreading** across multiple collision events
- **mkFit is integrated in CMS framework (CMSSW), with a single threaded application it is 6x faster**
  - mkFit compiled using icc and AVX-512 extensions
  - track building with mkFit is faster than CMSSW track building
  - integration of mkFit in CMS workflows is currently under investigation



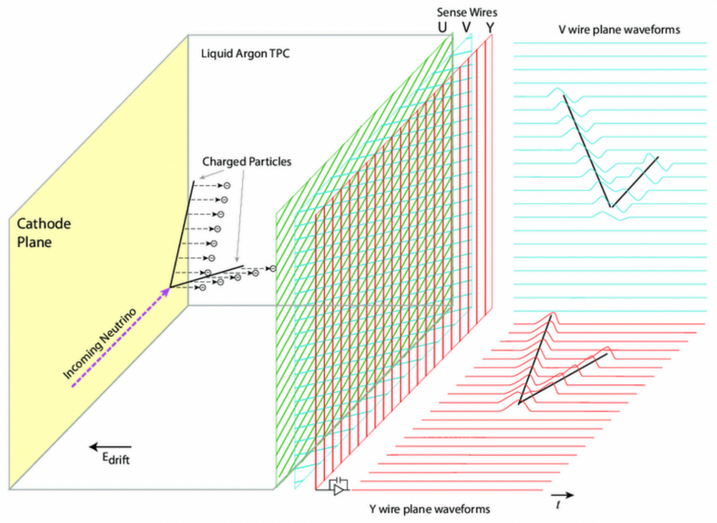


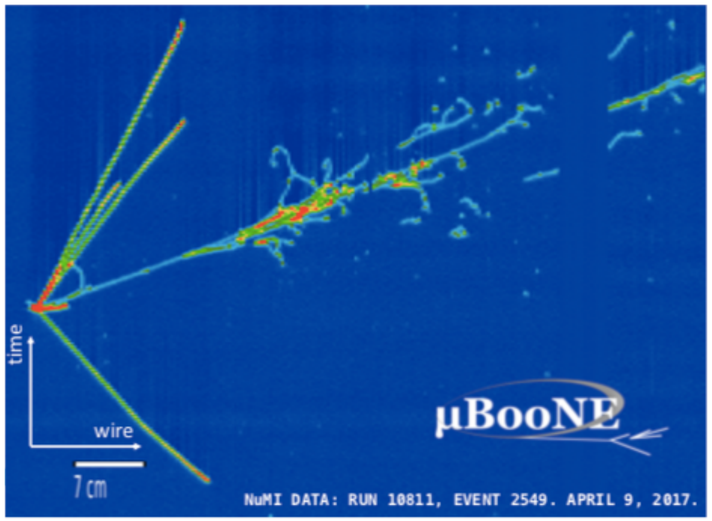
thanks to Giuseppe Cerati for slide contents

HEP Event Reconstruction: 2. Reconstruction for LArTPC ν Experiments

Reconstruction in LArTPC experiments is challenging due to the unknown interaction point, many possible topologies, noise, contamination of cosmic rays

- Particle tracking in LArTPC detectors:
  - charged particles produced in neutrino interactions ionize the Argon
  - ionization electrons drift in an electric field towards anode planes
  - sense wires detect the charge
- It takes O(minutes)/event for reconstruction in MicroBooNE
  - ICARUS detector is ~5x bigger
  - DUNE Far Detector will be O(100x) bigger
- Hit finding involves identifying pulses and determining their width and position (significant fraction of total time of workflow)
- Wires can be processed independently and thus hit finding lends itself to parallelism





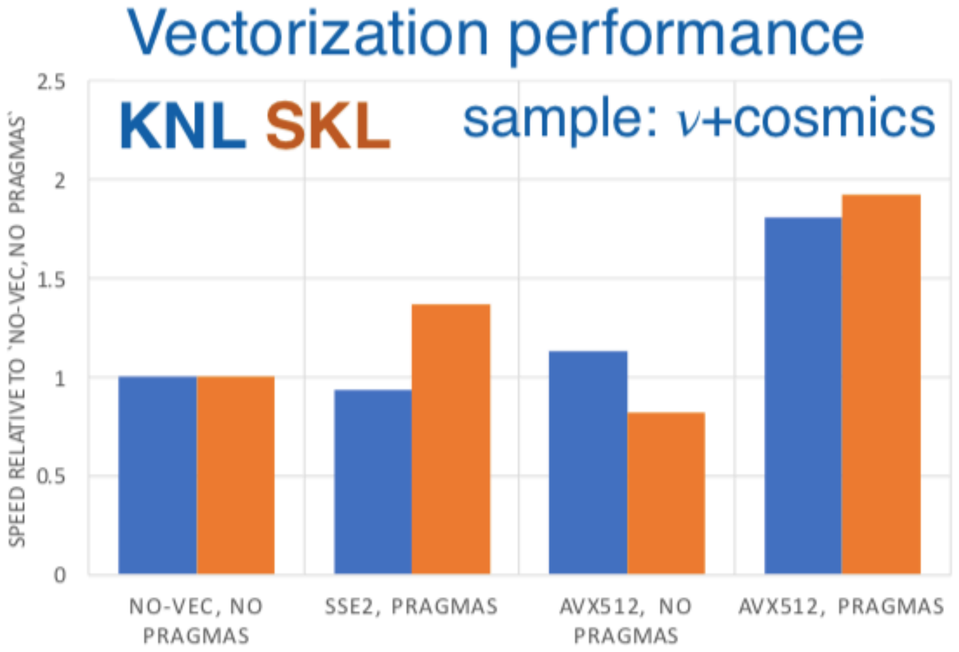
Fermilab

thanks to Giuseppe Cerati for slide contents

HEP Event Reconstruction: 2. Reconstruction for LArTPC ν Experiments

Parallelization of HitFinding Algorithm

- **Hit finder algorithm has been successfully parallelized:**
  - for more details see https://indico.cern.ch/event/868940/contributions/3814728/
- **Results:**
  - vectorization gives ~2x speedup when compiling with icc+AVX512 (both Skylake Gold and KNL)
  - OpenMP **multi-threading** shows near ideal scaling at low thread counts, with speedups increasing **up to 30x** (95x) for 80 (240) threads on Skylake Gold (KNL)
  - physics output nearly identical to original algorithm
  - new version integrated in the experiment codebase (larsoft) and **adopted by ICARUS and DUNE**
  - **Single-threaded application up to 10x faster** than previous version - significant impact on reco time of experiments!



Fermilab

thanks to Giuseppe Cerati for slide contents

HEP Data Analytics on HPC (a SciDAC Project)

Goal: to accelerate HEP **analysis** on HPC platforms

PandAna: a Python library for network analysis used to provide an easy-to-use environment for scalable high-level HEP analysis on HPC.

- Demonstrated a **scalable parallelization of an analysis code** from NovA by replacing serial I/O with parallel I/O
- Allows **existing analysis code developed by experimenters** to be deployed at HPC sites for processing of large datasets
- HDF5 for fast parallel reading of large amounts of date
- Can use Python and popular Python data science tools (numpy, pandas)
- Introducing the “tidy data” analysis model to HEP, using large data matrices and distributed data parallelism
  - use MPI for distributed parallelism
  - parallelism in user code is implicit

HEP Data Analytics on HPC (a SciDAC Project)

Goal: to accelerate HEP **analysis** on HPC platforms

PandAna: a Python library for network analysis used to provide an easy-to-use environment for scalable high-level HEP analysis on HPC.

- Demonstrated a **scalable parallelization of an analysis code** from NovA by replacing serial I/O with parallel I/O
- Allows **existing analysis code developed by experimenters** to be deployed at HPC sites for processing of large datasets
- HDF5 for fast parallel reading of large amounts of date
- Can use Python and popular Python data science tools (numpy, pandas)
- Introducing the “tidy data” analysis model to HEP, using large data matrices and distributed data parallelism
  - use MPI for distributed parallelism
  - parallelism in user code is implicit

first professions female
Pan flute player

Leopard Yellow Seamless
Facemask Bandana

HEP Data Analytics on HPC (a SciDAC Project)

Goal: to accelerate HEP **analysis** on HPC platforms

PandAna: a Python library for network analysis used to provide an easy-to-use environment for scalable high-level HEP analysis on HPC.

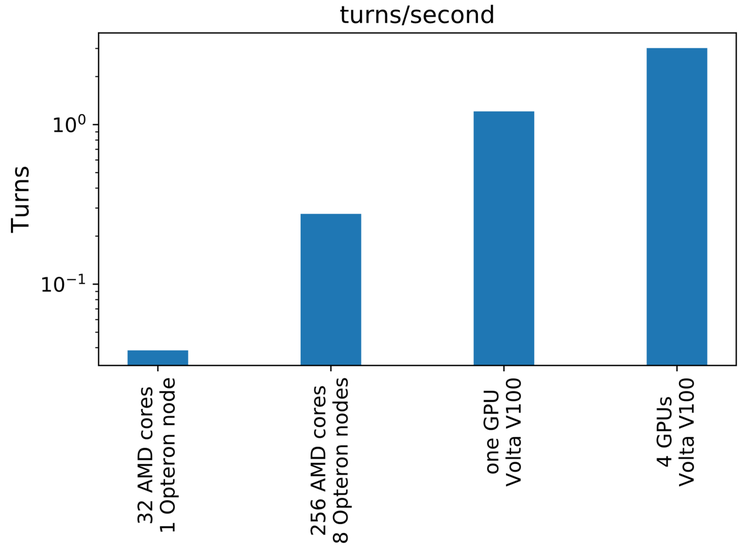
- Demonstrated a **scalable parallelization of an analysis code** from NovA by replacing serial I/O with parallel I/O
- Allows **existing analysis code developed by experimenters** to be deployed at HPC sites for processing of large datasets
- HDF5 for fast parallel reading of large amounts of date
- Can use Python and popular Python data science tools (numpy, pandas)
- Introducing the “tidy data” analysis model to HEP, using large data matrices and distributed data parallelism
  - use MPI for distributed parallelism
  - parallelism in user code is implicit

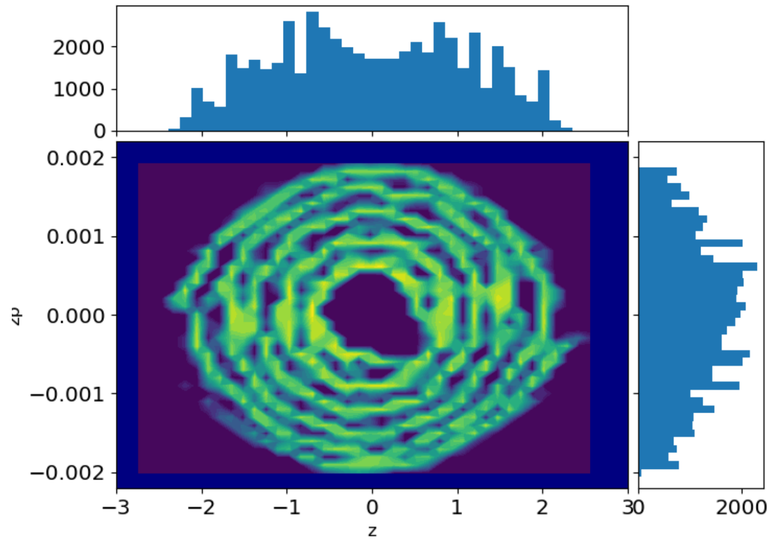
first professions female
Pan flute player

Accelerator Modeling

The Accelerator Modeling group develops and runs our locally produced Synergia modeling framework to numerically simulate beam physics processes in our accelerators.

- Synergia upgraded to use GPU and CPU computing with a single code-base
  - achieve a huge performance boost
  - future large computing resources will be primarily GPU based
- Simulations to support PIP-II
  - ongoing work to validate injection painting from the PIP-II Linac into the Booster
- Simulations of active space charge compensation with electron lenses
  - may be key to future higher intensity accelerators
  - have demonstrated the conditions required for partial space charge compensation for the first time in detailed particle simulations





Fermilab

thanks to Eric Stern for slide contents

Additional Ongoing Projects

- **CMS**
  - Elastic Analysis Facility: new analysis paradigm based on industry technologies with aim to provide improved computing resources to analyzers for interactive jobs
  - long term storage: the future of tape and how to safely store exabytes of data
  - networking: how to manage network flows and move exabytes into processing resources like HPC sites
- **Spack: package manager to replace antiquated UPS package distribution system**
  - supports combinatorial versioning like UPS (multiple packages, versions, OS-es and build variations)
  - lots of recipes for multithreading libraries
  - expect to be working with experiments to migrate their software distribution in the near future

Additional Ongoing Projects

- **Rucio: next generation data management framework for FNAL experiments**
  - open-source software framework that provides scientific collaborations with the functionality to organize, manage, and access their data at scale
  - originally developed to meet the requirements of ATLAS
  - data can be distributed across heterogeneous data centers at widely distributed locations
  - SCD is contributing to the codebase in bug fixes and features
- **Big Data Express: "aims to provide Schedulable, Predictable, and High-performance data transfer service for DOE large-scale science computing facilities"**
- **…**

Summary

- HEP experiments will be producing **more data than ever** as the size and granularity of detectors increases, and beam intensities increase.
- **High Performance Computing** is likely going to supersede High Throughput Computing as the paradigm for HEP computing.
- High Performance Computing Centers are going to play a major role in HEP computing in the future - most of the computing will be done at these centers.
- **Technologies** (hardware, software tools, data movement and storage tools, etc.) **change every ~5 years** at these centers.
- In order to deal with the changing computing and data landscapes, **many substantive changes need to be made**.
- SCD is working hard to meet these challenges (e.g. in reconstruction, analysis, accelerator modeling, networking, data handling,…)

Backup

HEPCloud Status

- Experiments (CMS, DUNE, Mu2e, NOvA) are already using HEPCloud for some production computing on Cori at NERSC
- Development is ongoing:
  - **Integration of Leadership Computing Facilities into the system is planned**
  - Development of a Global DashBoard for improved monitoring for both operations and users (via Landscape?)
  - Creation of a packaging and deployment infrastructure
  - Development of new resource provisioning methodologies (e.g.will need to support MPI and pipelined workflows)

HEP Data Analytics on HPC: HEPnOS

HEPnOS is built using Mochi, a framework for developing specialized data services for use in high performance computing. Mochi allows use of state of the art libraries while providing convenient APIs to scientists

- **Goals:**
  - manage physics event data from simulation and experiment through multiple phases of analysis
  - speed up access by retaining data in the system throughout analysis process
- **Properties:**
  - read in data at start of run and write results to persistent storage at the end of a campaign
  - hierarchical namespace matching physics concepts (datasets, runs, subruns)
  - C++ API (serialization of C++ objects)