HPC for HEP: Friend or Foe?

Salman Habib
CPS and HEP Divisions
Argonne NATIONAL LABORATORY

Basic Arithmetic:

- Current HEP computing worldwide is roughly a million cores (~few Pflops); US share is ~1Pflop
- This might increase to a few PFlops by the mid-2020's; HL-LHC requirement is (very) roughly order of magnitude more (but could be less)
- By 2022, the LCFs and NERSC will have ~Eflops of compute power, by the mid 2020's, tens of Eflops of compute power — ***4 orders of magnitude more than all US HEP computing combined***
- HEP usually gets ~10% of ASCR compute capability, still have 3 orders of magnitude; let us say LHC experiments get 10% of this, still have 2 orders of magnitude, let's say usage efficiency is not great, ***still may have 10X the equivalent of the total compute requirement***

Dose of Reality:

- Next-gen architectures (GPUs, ARM, etc.) require handling ***massive (local) concurrency*** and are ***difficult to program*** plus the ***future is very hard to predict***
- HEP experiment software — like many others — not up to the task as yet (but some encouraging early results)
- ECP project focuses on HPC codes and these are making solid progress (but note these **are** HPC codes)
- HEP experiment software is in the millions of lines of code, but ***most of it probably does not need to be refactored***
- DOE/SC HPC codebase is probably O(100M) lines of code; include NNSA, probably a billion — next-gen computing is a ***problem faced by everyone***
- Many other ***data-intensive use cases*** exist on the HL-LHC timescale (light sources, genomics, distributed sensors, —)
- Main issues — 1) exploit ***concurrency*** (SIMD/SIMT etc.) but 2) be ***portable*** — able to run on all systems and get the same answer (nontrivial)
- Secondary issues are technical (I/O, storage, edge services, temperamental HPC systems) and political, but surmountable

Example System: Aurora

Argonne NATIONAL LABORATORY

Public Specs:

- Performance (sustained): >1EF (double precision)
- System interconnect: Cray Slingshot
- Compute node: x86 CPUs + Intel GPUs
- CPU-GPU interconnect: PCIe
- System Memory: >10PB
- Storage: >230PB at >25TB/s
- # Cabinets: >100
- Programming Languages/Models: Intel OneAPI, Fortran, C, C++, UPC, Co-array Fortran, Python, MPI, OpenCL, OpenMP 5.x, DPC++/SYCL
- Frameworks: TensorFlow, PyTorch, Scikit-learn, GraphX, Intel DAAL, Intel MKL-DNN

HEP ECP project example:

- ExaSky — extreme-scale cosmological simulations, ~0.5M lines of code, two main codes (HACC and Nyx), one large-scale analysis framework
- HACC has about 300K lines of code, 95% are machine-independent C++/MPI
- Typical HACC performance is around ~50+% of peak on all DOE systems, ***independent of architecture***
- Note: "HPC application and software development is a ***contact sport***" — Doug Kothe

AI-friendly!

One cabinet is ~10PFlops (Cf. US HEP HL-LHC computing requirement naively scaled up) and this will be ***already obselete in the HL-LHC era***

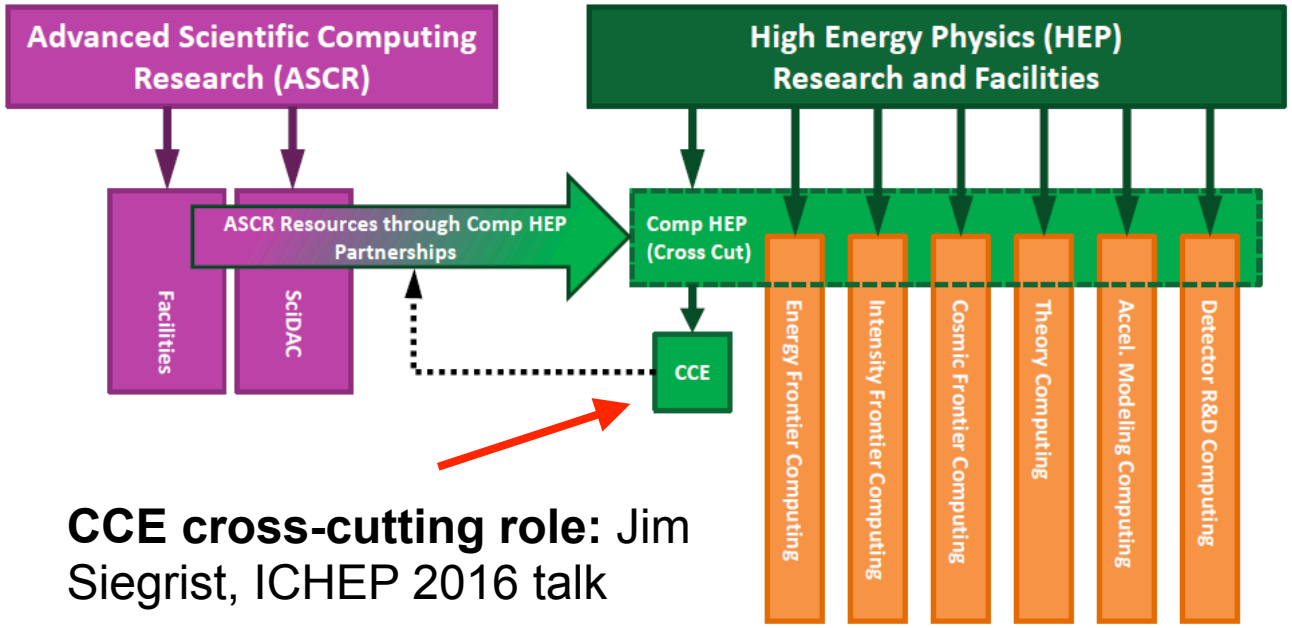
HEP-CCE: A HEP link to ASCR

https://press3.mcs.anl.gov/hepfce/

Argonne NATIONAL LABORATORY

CCE:

- Consists of HEP and ASCR researchers at HEP laboratories (***very*** partially supported)
- Main role is to develop HPC tools/capabilities to aid HEP science
- Helped bring containers to HPC systems; jointly worked with ESnet on Petascale Transfer project, run summer student programs, ported HEP code on ALCF systems and at NERSC, sponsored hackathons and workshops (event generators, IO), etc.
- Current proposal aims to study use of ASCR HPC systems ***for all*** HEP frontiers particularly experiments taking data starting from 2020 onwards (ATLAS, **CMB-S4**, CMS, **DESI**, **DUNE**, **LSST DESC**, —)
- Primary focus is on **1)** pilot projects on concurrency/ offloading leveraging ongoing work by experiments, **2)** data model/structure issues and IO, **3)** event generation, **4)** complex distributed workflows on HPC systems (mostly CF) — first two are (by far) the most important



CCE cross-cutting role: Jim Siegrist, ICHEP 2016 talk

ATLAS Connection (possible):

- Doug Benjamin
- Paolo Calafiura
- Taylor Childers
- Charles Leggett
- Peter van Gemmeren
- —

Back-Up Info

Argonne NATIONAL LABORATORY

PPS

- Data structures that support SIMD/SIMT parallelization targeting CPU/vector units and GPUs
- Efficient communication between CPUs and GPUs
- Memory layout optimization to enable bacth operations spanning multiple events
- Potential use cases: FastJet, Track seeding, LAr TPC data preparation
- Investigate Kokkos, RAJA, OpenMP, DPC++/SYCL, etc. as performance portability models (also Python/Numba)

Timescales

- FY20/Q1 — US ATLAS/CMS ops program HL-LHC R&D strategic plan
- FY20/Q1 — DUNE software and computing CDR
- FY20/Q2-Q3 — ATLAS/CMS interim R&D statements
- FY21/Q3 — CMS High Level Trigger TDR
- FY22/Q1 — ProtoDUNE-II beam operations, DUNE Software and Computing TDR
- FY22/Q3 — WLCG and ATLAS/CMS Software and Computing TDRs