Computing for Future HEP Analysis

James Amundson, Fermilab

2019 Meeting of the Division of Particles & Fields of the American Physical Society

August 2, 2019

Computing for Future HEP ~~Analysis~~

James Amundson, Fermilab

2019 Meeting of the Division of Particles & Fields of the American Physical Society

August 2, 2019

Future of Computing

~~Computing for Future HEP Analysis~~

James Amundson, Fermilab

2019 Meeting of the Division of Particles & Fields of the American Physical Society

August 2, 2019

Technology for High Energy Physics

DPF talks today

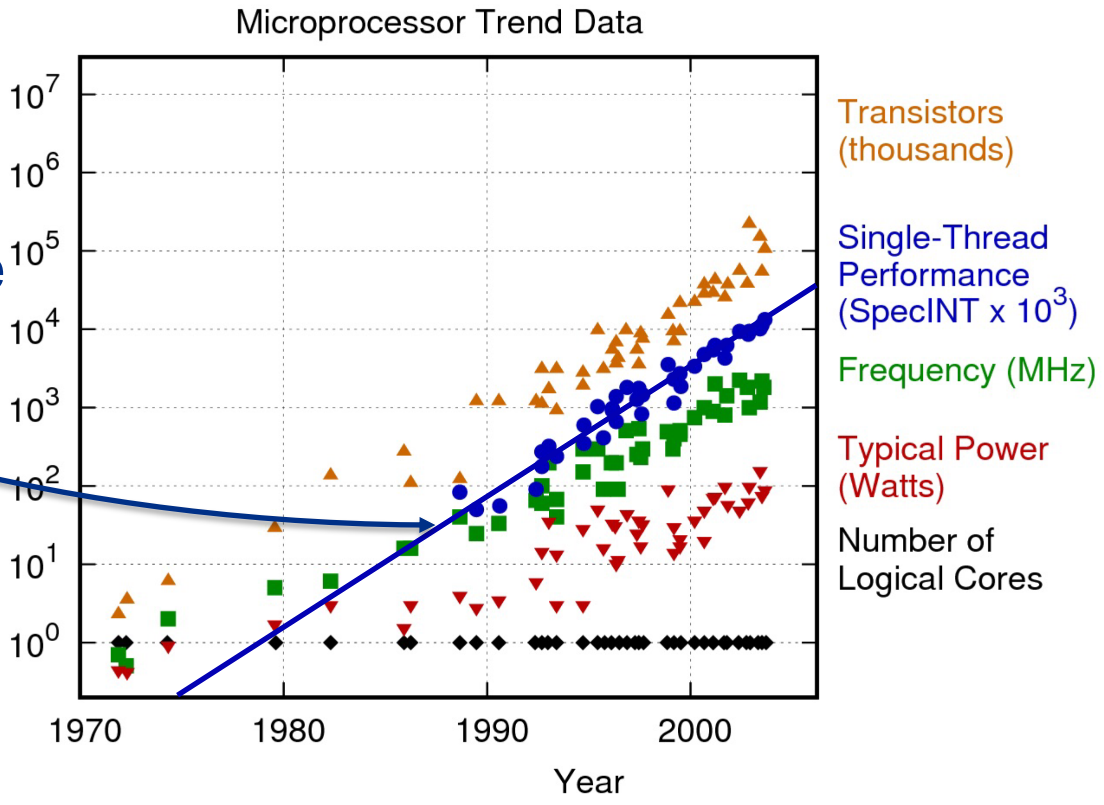
- **Applications of Detector Technology**
  - Detector technology:
    - Developed by HEP physicists and engineers
    - Strong overlaps with other sciences, industrial applications
- **Advanced Accelerator Concepts**
  - Particle accelerator technology:
    - Developed by accelerator physicists and engineers
      - Close ties to HEP (training, proximity, etc.)
    - Applications in other sciences, industry
      - See, e.g., BRN on Compact Accelerators for Security and Medicine
- **Computing for Future HEP Analysis**
  - Computing technology:
    - Used every day by everyone* in the industrialized world
      - * American Academy of Pediatrics recommendation: "*For children younger than 18 months, avoid use of screen media other than video-chatting.*"
        - No recommendations about either detector or particle accelerator technology

HEP is not the driving force behind advances in computing technology

- Very large scale computing is no longer unique to HEP
  - Large data sets (larger than HEP)
    - Google, Facebook, etc.
  - “Big Data” analysis (HEP is still probably the largest)
    - Already a cliché
  - Large-scale simulation
    - More on that soon…
  - Machine Learning
    - More on that, too
- Helpful for HEP in the long run
  - We no longer have to invent everything ourselves
  - Students and postdocs have the opportunity to learn skills that are useful outside of HEP

"Moore's Law" – the good old days

speedup: 49x / decade



https://www.karlrupp.net/2018/02/42-years-of-microprocessor-trend-data/

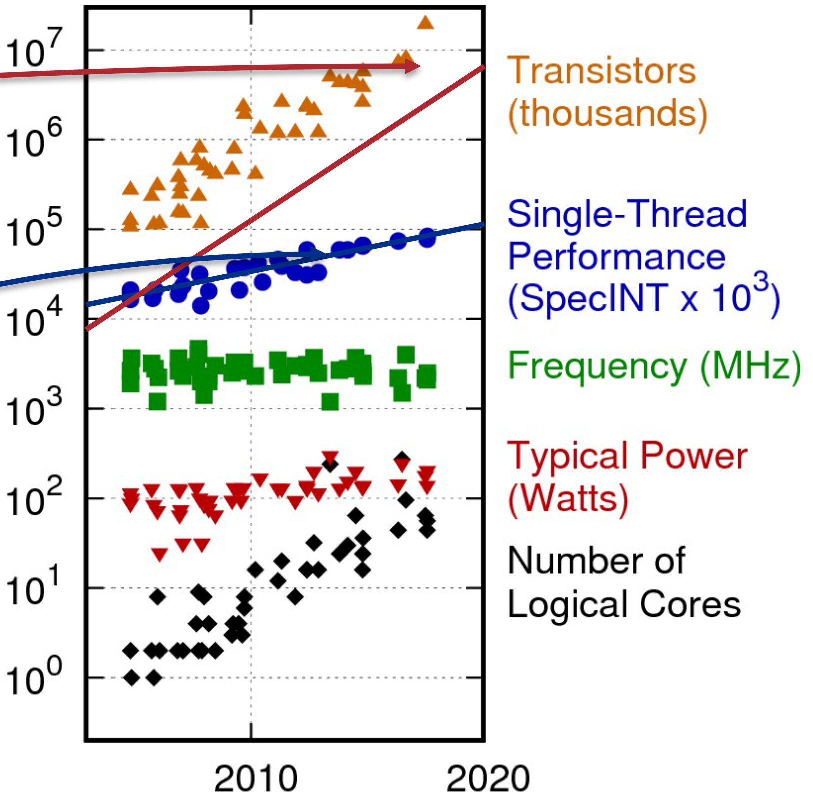
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

Fermilab

“Moore’s Law” – recent times

old trend

new speedup: 3.3x / decade



where’s my flying car?

62x

Fermilab

Changes are happening in computing

- Terminology
  - High Throughput Computing (HTC)
    - "Linux farms"
      - Commodity CPUs, x86-64
      - Commodity networking

  - High Performance Computing (HPC)
    - "Supercomputers"
      - Variety of CPUs, some x86-64, some not
      - Low-latency, high-speed networking

HPC for HEP

- Nearly all HEP computing has been HTC until recently
  - Low-latency networks are mostly wasted on HEP
- HPC is becoming more attractive
  - Newer HPC machines focus on energy efficiency
    - Increasingly important as computing needs grow
  - HPC is increasingly the leading edge of commodity technology
    - Advances in computing architecture come to HPC first
      - Important in changing times
  - DOE has made a large investment in HPC
    - Exascale!
      - $2B project to build two computers capable of 10^{15} floating point operations/second

Exascale is all about power

Previous generation supercomputer: Mira (10 Petaflop supercomputer, 4 MW)

1 Exaflop =

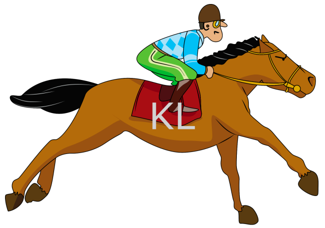
100 x

+

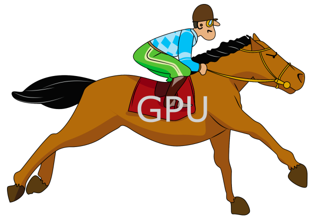
Goal of Exascale program is 20-40 MW

Fermilab

Competing technologies for Exascale



- Intel Knights- Series
  - Many (hyper-)threads/CPU (~256)
  - Wide Single Instruction Multiple Data (SIMD) (aka Vector)
    - AVX-512: 8 doubles
      - vs AVX2 (4 doubles) in commodity machines
  - Most similar to commodity machines (in principle)



- GPU
  - NVIDIA GPUs most popular
  - CPU/GPU memory distinction
  - CUDA (NVIDA-specific) most popular programming model

- Both architectures: Work best with simple, dense memory structures, simple loops
  - Arrays of structures: bad
  - Structures of arrays: good

Winners of the Race

- Architectures for Exascale machines have been announced
  - x86_64 + GPUs (!)
    - Not NVIDIA GPUs
      - CUDA (probably) not native (!)
- ALCF (Argonne)
  - Aurora
    - 2021
    - > 1 Exaflop
    - Intel
- OLCF (Oak Ridge)
  - Frontier
    - 2021
    - > 1.5 Exaflop
    - AMD

Computing for Future HEP Challenges

Liz Sexton-Kennedy
Fermilab CIO

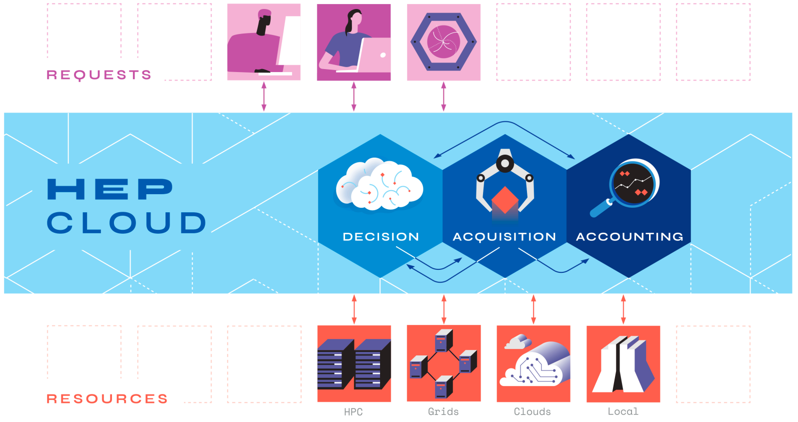
- The LHC experiments, Belle II and DUNE face the same challenges
  - HEP software must evolve to meet these challenges
  - Need to exploit all the expertise available, inside and outside our community, for parallelization
  - New approaches needed to overcome limitations in today's code
- Cannot afford any more duplicated efforts, it is unsustainable
  - Currently each experiment has its own solution for almost everything (framework, reconstruction algorithms, …)
  - It is not clear how much can be shared however today's situation means there are lots of opportunities.
- Even if we had the money for all of this duplication we do not have the requisite expertise to do all of the work necessary for all experiments!
  - A number of people recognized this a few years ago and formed the HEP Software Foundation

Computing for Future HEP Strategy

- Plan for a heterogeneous computing environment
  - HTC grid resources
    - with and without GPUs
  - HPC resources
    - Especially Exascale
  - Commercial cloud resources
- Continue leadership in *active* mass storage at the labs
  - Most current HPC mass storage is archival only
- Leverage common development efforts within HEP
- Leverage developments in machine learning
- Leverage tools from outside of HEP
  - Give students and postdocs skills they can use in industry
  - Support from a much broader community
    - Answers can be found on the web, not just down the hall
  - Etc., etc.

Planning for a Heterogeneous Computing Environment

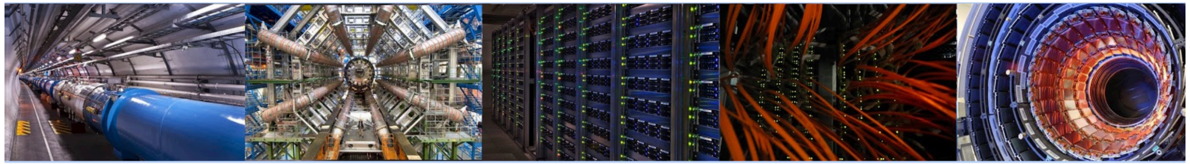
- Example project: HEPCloud
  - https://hepcloud.fnal.gov/
  - HEPCloud provides a universal portal to computing resources
    - Conventional
    - Cloud
    - HPC
  - First production use at Fermilab started this spring
- Example project: SCAILFIN
  - See yesterday's presentation by Mike Hildreth



Large-scale HPC deployment of Scalable CyberInfrastructure for Artificial Intelligence and Likelihood Free Inference (SCAILFIN)

Common Development Efforts within HEP: IRIS-HEP

- See Peter Elmer's Talk yesterday



Institute for Research and Innovation in Software for High Energy Physics (IRIS-HEP)

PI: Peter Elmer (Princeton), co-PIs: Brian Bockelman (Morgridge Institute), Gordon Watts (U.Washington) with UC-Berkeley, University of Chicago, University of Cincinnati, Cornell University, Indiana University, MIT, U.Michigan-Ann Arbor, U.Nebraska-Lincoln, New York University, Stanford University, UC-Santa Cruz, UC-San Diego, U.Illinois at Urbana-Champaign, U.Puerto Rico-Mayaguez and U.Wisconsin-Madison

http://iris-hep.org

IRIS-HEP was funded as of 1 September, 2018, and is currently ramping up activities

OAC-1836650

Common Development Efforts within HEP: IRIS-HEP

IRIS-HEP

Sustainable Software R&D objectives

1) Development of **innovative algorithms** for data reconstruction and triggering;

2) Development of highly performant **analysis systems** that reduce "time-to-insight" and maximize the HL-LHC physics potential; and

3) Development of **data organization, management and access systems** for the community's upcoming Exabyte era.

4) Integration of software and scalability for use by **the LHC community on the Open Science Grid**, the Distributed High Throughput Computing infrastructure in the U.S.

IRIS-HEP funded as a 5 year project from 1 Sep, 2018

Intellectual Hub for the HEP Community

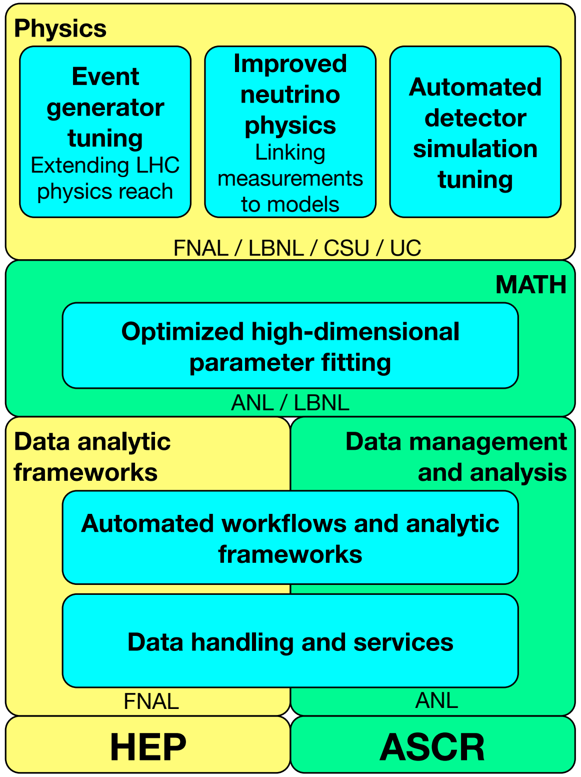
The plan for IRIS-HEP reflects a community vision developed by an international community process organized by the HEP Software Foundation (https://hepsoftwarefoundation.org). The S2I2-HEP conceptualization project (http://s2i2-hep.org) derived a Strategic Plan from the community roadmap which would leverage the strengths of the U.S. university community. IRIS-HEP aims to function as an intellectual hub for the national and international HEP community, through training, community workshops and the development of wider collaborations with the larger computer and data science communities.

Common Development Efforts within HEP: SciDAC Partnerships

- DOE Scientific Discovery through Advanced Computing (SciDAC) Partnerships between HEP and ASCR
  - Accelerating HEP Science: Inference and Machine Learning at Extreme Scales
    - Cosmology, collider and neutrino experiments
    - PI: Habib (ANL)
  - Community Project for Accelerator Science and Simulation 4 (ComPASS4)
    - Accelerator simulation
    - PI: Me (FNAL)
  - HEP Data Analytics on HPC
    - PI: Kowalkowski (FNAL)
  - HEP Event Reconstruction with Cutting Edge Computing Architectures
    - PI: Cerati (FNAL)
  - HPC Framework for Event Generation at Colliders
    - PI: Hoeche (FNAL)

SciDAC: HEP Data Analytics on HPC

- Physics component:
  - Participate in getting new physics results out
  - LHC and neutrino experiments
- Accelerate targeted HEP analysis on HPC platforms
  - Working within experiment code and operations
  - In some cases by orders of magnitude
- Transform how physics tasks are carried out using ASCR tools and techniques
  - Parallel storage and access for experiment data
  - Data-parallel programming and advance mathematical techniques for HEP analysis



Physics
Event generator tuning
Extending LHC physics reach
Improved neutrino physics
Linking measurements to models
Automated detector simulation tuning
FNAL / LBNL / CSU / UC
MATH
Optimized high-dimensional parameter fitting
ANL / LBNL
Data analytic frameworks
Data management and analysis
Automated workflows and analytic frameworks
Data handling and services
FNAL
ANL
HEP
ASCR

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

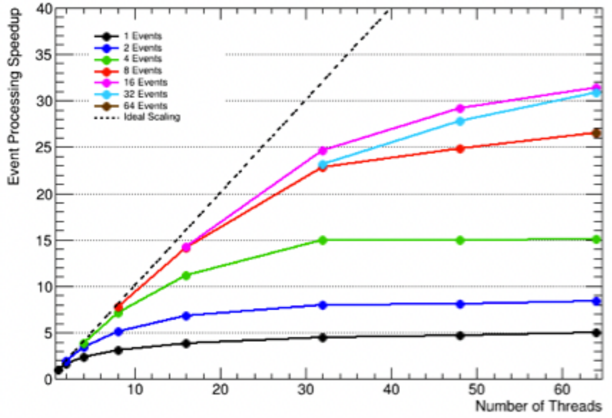
Fermilab

SciDAC: HEP Event Reconstruction with Cutting Edge Computing Architectures

Project goals

- Accelerate HEP event reconstruction using modern parallel architectures.

- Focus on two areas:
  - Novel parallel algorithm for charged particle **tracking in CMS**
    - Cornell/FNAL/UCSD/UOregon/Princeton collaboration
    - Non-SciDAC funding from NSF IRIS-HEP and from USCMS
  - Pioneer similar techniques for **reconstruction in LArTPC detectors**



- Goals of the project are the following:
  1. Identify key algorithms for the physics outcomes of each experiment, and that also are dominant contributors to their reconstruction workflows
  2. Characterize and re-design the algorithms to make efficient usage of parallelism, both at data- and instruction-level
  3. Deploy the new code in the experiments' framework
  4. Explore execution on different architectures and platforms

Common Development Efforts within HEP: HEP CCE

- HEP Center for Computing Excellence
  - Funded by DOE CompHEP
  - Collaboration between four labs
    - Argonne
    - Fermilab
    - Lawrence Berkeley
    - Brookhaven
  - Focus on HEP utilization of DOE Leadership Class Facilities (LCFs)
    - Argonne LCF
    - Oak Ridge LCF
  - Next-generation project coming soon

Leveraging Developments in Machine Learning

- Machine Learning is taking off everywhere
- HEP is no exception
  - See many of the talks in the Computing and Analysis Tools parallel session
    - Forgive me for not listing them all
- Most machine learning applications are using external tools
  - Tensorflow, Pytorch, etc.
- Adaptations for new architectures come for “free”
  - Huge advantage from leveraging tools from outside HEP

Computing for Future HEP Analysis (Finally)

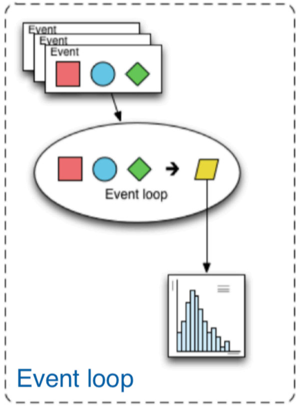
- Two examples of progress in HEP analysis
  - The Coffea project
    - See Nick Smith’s talk on Tuesday
    - Collaboration with IRIS-HEP
    - Ticks multiple boxes
      - Heterogeneous computing environment, common solutions, machine learning, external tools
  - HDF5-based NOvA analysis
    - Part of the SciDAC HEP Data Analytics project
    - Ticks multiple boxes
      - Heterogeneous computing environment, external tools

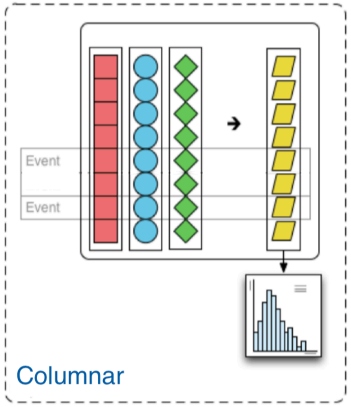
Coffea: Columnar analysis

What is columnar analysis?

- Event loop analysis:
  - Load relevant values for a specific event into local variables
  - Evaluate several expressions
  - Store derived values
  - Repeat (explicit outer loop)

- Columnar analysis:
  - Load relevant values for many events into contiguous arrays
    - Nested structure (array of arrays) → flat content + offsets
      - This is the ROOT TTree internal data structure!
  - Evaluate several **array programming** expressions
    - Implicit *inner* loops
  - Store derived values

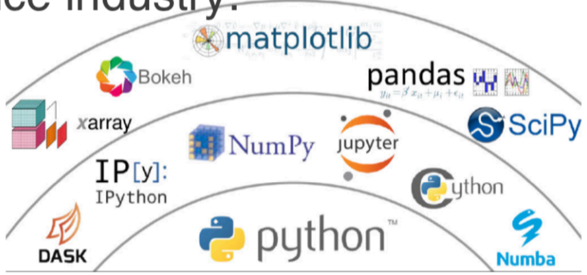




Coffea: external tools

Columnar analysis ecosystem

- Array programming is a mainstay of data science industry:
  - **Data structure**: numpy, arrow
  - **Data exploration**: pandas, matplotlib, …
  - **Algorithms**: numpy, numba, …
  - **Machine Learning**: tensorflow, pytorch, …
  - **Big Data**: spark, dask, …



NOvA Analysis

- Comparisons done using neutrino data from the NOvA Experiment
  - NOvA produces both ROOT and HDF5 version of their analysis ntuples
    - HDF5 version are on average 14% smaller than ROOT representation
  - NOvA also has two separate analysis frameworks for examining the data
    - CAFAna based on ROOT TTree with C++ class objects
    - PandAna based on HDF5 Tables with Pandas dataframes
  - Tests performed on same platform with data files staged locally
    - Examines 117482 cosmic ray background events and reads the number of hits in the events to compute a sum
    - ROOT Based Analysis using pyROOT following standard NOvA Tutorial Pattern (2019 Tutorial set)
    - Python Analysis reading from HDF5 and using numpy and h5py

NOvA Analysis: ROOT vs. HDF5

Root

```
#!/usr/bin/env python
import sys
from ROOT import TFile, TTree, gSystem
import time

# Load NOvA Experiment Class Definitions
gSystem.Load("$SRT_PUBLIC_CONTEXT/lib/Linux2.6-GCC-
maxopt/libStandardRecord_dict.so")
from ROOT import caf

file= TFile("test.root")

tree = file.Get("recTree")
rec = caf.StandardRecord()
tree.SetBranchAddress("rec",rec)

start = time.time()
nentries = tree.GetEntries()
total_nhit = 0
for i in range(nentries):
    tree.GetEntry(i)
    total_nhit += rec.slc.nhit
stop = time.time()

delta = stop - start
print delta, total_nhit
```

Execution Time: 80.8s +- 1.3s

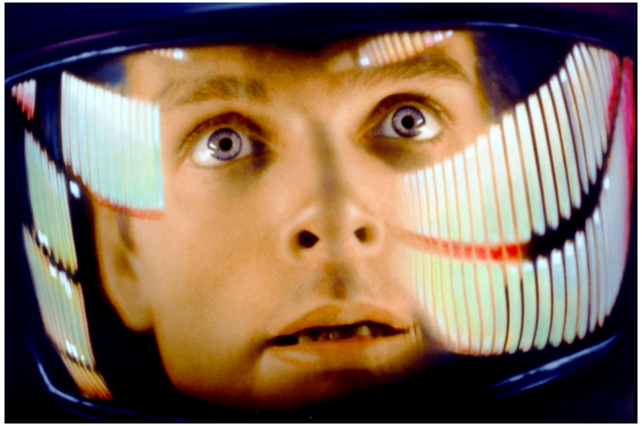
HDF5

```
#!/usr/bin/env python
import numpy as np
import h5py as h5
import time

f = h5.File("test.h5")
start = time.time()

nhit = f["rec.slc"]["nhit"][:]
total_nhit = np.sum(nhit)
stop = time.time()
delta = stop - start
print delta, total_nhit
```

Execution Time: 0.018s+-0.002s

Conclusions

- We have seen the future of computing, and it is very different from the past
- We will meet the computing challenges for HEP by
  - Embracing heterogeneous computing
    - HTC
    - HPC
  - Continuing leadership in mass storage at the labs
  - Leveraging machine learning
  - Leveraging community computing efforts
  - Leveraging outside tools



My God, it’s full of GPUs