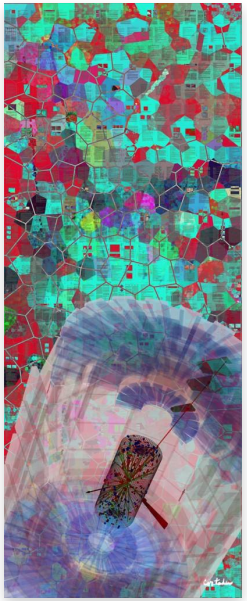
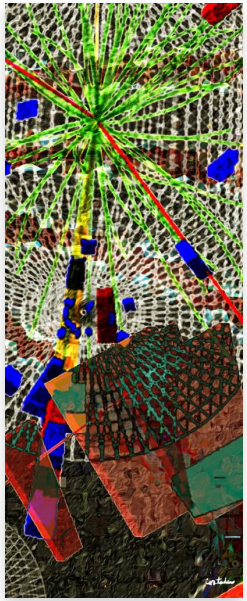
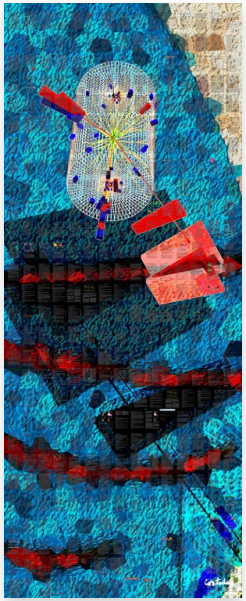
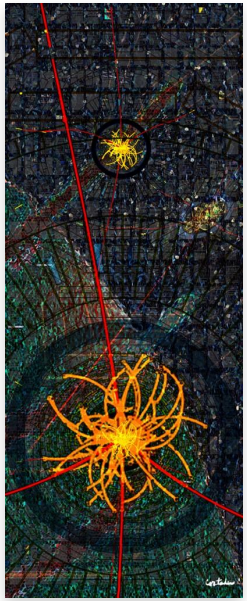
U.S. CMS Operations Program

Computing and Software for Physics

Andrew Melo
Vanderbilt University
USCMS Undergraduate Summer Internship
June 7th, 2024

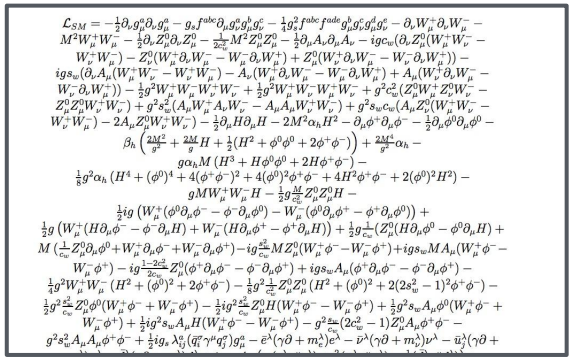
Overview

- Introduction
- Computing's role in CMS
- Challenges
- Accessing data and analyzing it

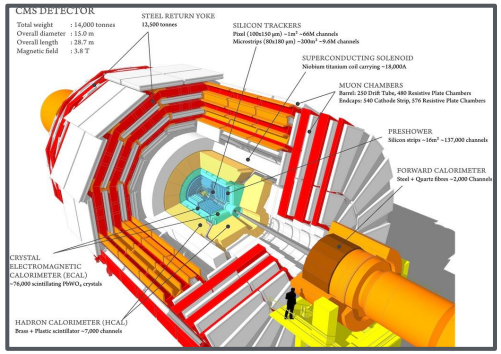
About Myself

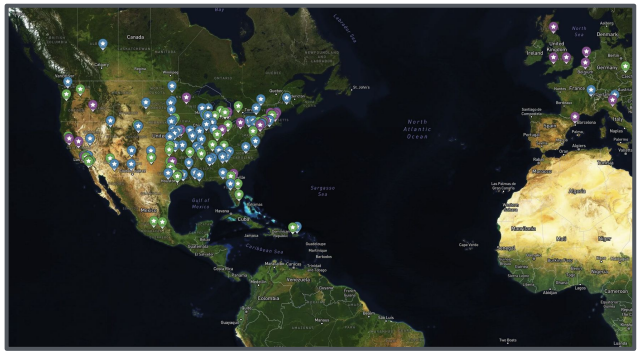
- Research Professor at Vanderbilt University
- Originally wanted to study Computer Science, ended up double-majoring in Physics
- Double-majored in Computer Science and Physics in undergrad
- PhD dissertation a search for evidence for dark matter w/CMS
- Active in CMS computing for nearly 15 years:
  - Contributed to workload management software
  - Maintain the CMS Facility @ Vanderbilt
  - Manages all U.S.-based CMS facilities

Computing's role in CMS



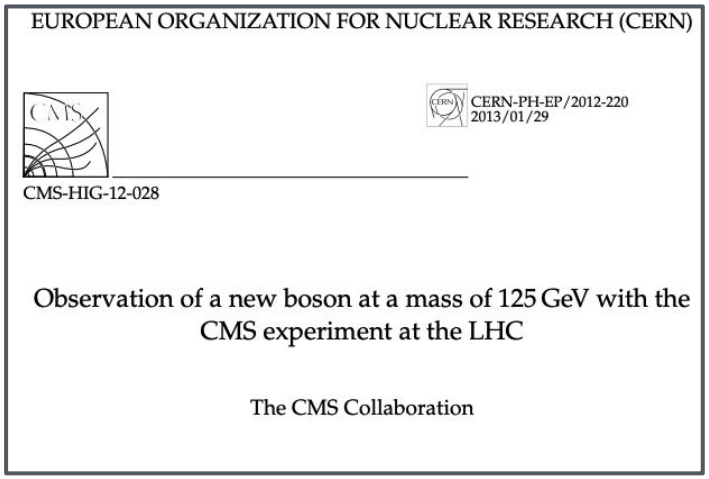
$$\mathcal{L}_{SM} = -\tfrac{1}{2}\partial_\nu g^a_\mu \partial_\nu g^a_\mu - g_s f^{abc}\partial_\mu g^a_\nu g^b_\mu g^c_\nu - \tfrac{1}{4}g_s^2 f^{abc} f^{ade} g^b_\mu g^c_\nu g^d_\mu g^e_\nu - \partial_\nu W^+_\mu \partial_\nu W^-_\mu -$$
$$M^2 W^+_\mu W^-_\mu - \tfrac{1}{2}\partial_\nu Z^0_\mu \partial_\nu Z^0_\mu - \tfrac{1}{2c_0^2} M^2 Z^0_\mu Z^0_\mu - \tfrac{1}{2}\partial_\mu A_\nu \partial_\mu A_\nu - igc_w(\partial_\nu Z^0_\mu(W^+_\mu W^-_\nu -$$
$$W^+_\nu W^-_\mu) - Z^0_\nu(W^+_\mu \partial_\nu W^-_\mu - W^-_\mu \partial_\nu W^+_\mu) + Z^0_\mu(W^+_\nu \partial_\nu W^-_\mu - W^-_\nu \partial_\nu W^+_\mu)) -$$
$$igs_w(\partial_\nu A_\mu(W^+_\mu W^-_\nu - W^+_\nu W^-_\mu) - A_\nu(W^+_\mu \partial_\nu W^-_\mu - W^-_\mu \partial_\nu W^+_\mu) + A_\mu(W^+_\nu \partial_\nu W^-_\mu -$$
$$W^-_\nu \partial_\nu W^+_\mu)) - \tfrac{1}{2}g^2 W^+_\mu W^-_\mu W^+_\nu W^-_\nu + \tfrac{1}{2}g^2 W^+_\mu W^-_\nu W^+_\mu W^-_\nu - g^2 c_w^2(Z^0_\mu W^+_\mu Z^0_\nu W^-_\nu -$$
$$Z^0_\mu Z^0_\mu W^+_\nu W^-_\nu) + g^2 s_w^2(A_\mu W^+_\mu A_\nu W^-_\nu - A_\mu A_\mu W^+_\nu W^-_\nu) + g^2 s_w c_w(A_\mu Z^0_\nu(W^+_\mu W^-_\nu -$$
$$W^+_\nu W^-_\mu) - 2A_\mu Z^0_\mu W^+_\nu W^-_\nu) - \tfrac{1}{2}\partial_\mu H \partial_\mu H - 2M^2\alpha_h H^2 - \partial_\mu\phi^+\partial_\mu\phi^- - \tfrac{1}{2}\partial_\mu\phi^0\partial_\mu\phi^0 -$$
$$\beta_h\left(\tfrac{2M^2}{g^2} + \tfrac{2M}{g}H + \tfrac{1}{2}(H^2 + \phi^0\phi^0 + 2\phi^+\phi^-)\right) + \tfrac{2M^4}{g^2}\alpha_h -$$
$$g\alpha_h M\,(H^3 + H\phi^0\phi^0 + 2H\phi^+\phi^-) -$$
$$\tfrac{1}{8}g^2\alpha_h\,(H^4 + (\phi^0)^4 + 4(\phi^+\phi^-)^2 + 4(\phi^0)^2\phi^+\phi^- + 4H^2\phi^+\phi^- + 2(\phi^0)^2H^2) -$$
$$gMW^+_\mu W^-_\mu H - \tfrac{1}{2}g\tfrac{M}{c_w^2}Z^0_\mu Z^0_\mu H -$$
$$\tfrac{1}{2}ig\,(W^+_\mu(\phi^0\partial_\mu\phi^- - \phi^-\partial_\mu\phi^0) - W^-_\mu(\phi^0\partial_\mu\phi^+ - \phi^+\partial_\mu\phi^0)) +$$
$$\tfrac{1}{2}g\,(W^+_\mu(H\partial_\mu\phi^- - \phi^-\partial_\mu H) + W^-_\mu(H\partial_\mu\phi^+ - \phi^+\partial_\mu H)) + \tfrac{1}{2}g\tfrac{1}{c_w}(Z^0_\mu(H\partial_\mu\phi^0 - \phi^0\partial_\mu H) +$$
$$M\,(\tfrac{1}{c_w}Z^0_\mu\partial_\mu\phi^0 + W^+_\mu\partial_\mu\phi^-) + W^-_\mu\partial_\mu\phi^+) - ig\tfrac{s_w^2}{c_w}MZ^0_\mu(W^+_\mu\phi^- - W^-_\mu\phi^+) + igs_wMA_\mu(W^+_\mu\phi^- -$$
$$W^-_\mu\phi^+) - ig\tfrac{1-2c_w^2}{2c_w}Z^0_\mu(\phi^+\partial_\mu\phi^- - \phi^-\partial_\mu\phi^+) + igs_wA_\mu(\phi^+\partial_\mu\phi^- - \phi^-\partial_\mu\phi^+) -$$
$$\tfrac{1}{4}g^2W^+_\mu W^-_\mu\,(H^2 + (\phi^0)^2 + 2\phi^+\phi^-) - \tfrac{1}{8}g^2\tfrac{1}{c_w^2}Z^0_\mu Z^0_\mu\,(H^2 + (\phi^0)^2 + 2(2s_w^2 - 1)^2\phi^+\phi^-) -$$
$$\tfrac{1}{2}g^2\tfrac{s_w^2}{c_w}Z^0_\mu\phi^0(W^+_\mu\phi^- + W^-_\mu\phi^+) - \tfrac{1}{2}ig^2\tfrac{s_w^2}{c_w}Z^0_\mu H(W^+_\mu\phi^- - W^-_\mu\phi^+) + \tfrac{1}{2}g^2s_wA_\mu\phi^0(W^+_\mu\phi^- +$$
$$W^-_\mu\phi^+) + \tfrac{1}{2}ig^2s_wA_\mu H(W^+_\mu\phi^- - W^-_\mu\phi^+) - g^2\tfrac{s_w}{c_w}(2c_w^2 - 1)Z^0_\mu A_\mu\phi^+\phi^- -$$
$$g^2s_w^2A_\mu A_\mu\phi^+\phi^- + \tfrac{1}{2}ig_s\lambda^a_{ij}(\bar{q}^\sigma_i\gamma^\mu q^\sigma_j)g^a_\mu - \bar{e}^\lambda(\gamma\partial + m^\lambda_e)e^\lambda - \bar{\nu}^\lambda(\gamma\partial + m^\lambda_\nu)\nu^\lambda - \bar{u}^\lambda_i(\gamma\partial +$$





After the collisions occur and are recorded by the detector, it's still not in a format usable by analyzers (you!). The data needs to be:

- Reconstructed
- Slimmed down for a format useful for analysis
- Transmitted to our "grid" of ~80 facilities in 50 countries
- Be processed by analysis software who transmits the final results to the end user



EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH (CERN)

CERN-PH-EP/2012-220
2013/01/29

CMS-HIG-12-028

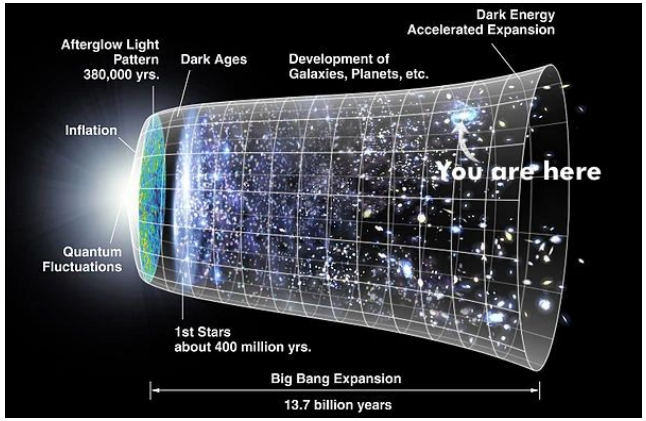
Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC

The CMS Collaboration

Scale of CMS Computing

"Space is big. You just won't believe how vastly, hugely, mind-bogglingly big it is. I mean, you may think it's a long way down the road to the chemist's, but that's just peanuts to space."

-Douglas Adams, The Hitchhiker's Guide to the Galaxy



CMS produces, stores and processes an enormous amount of data:

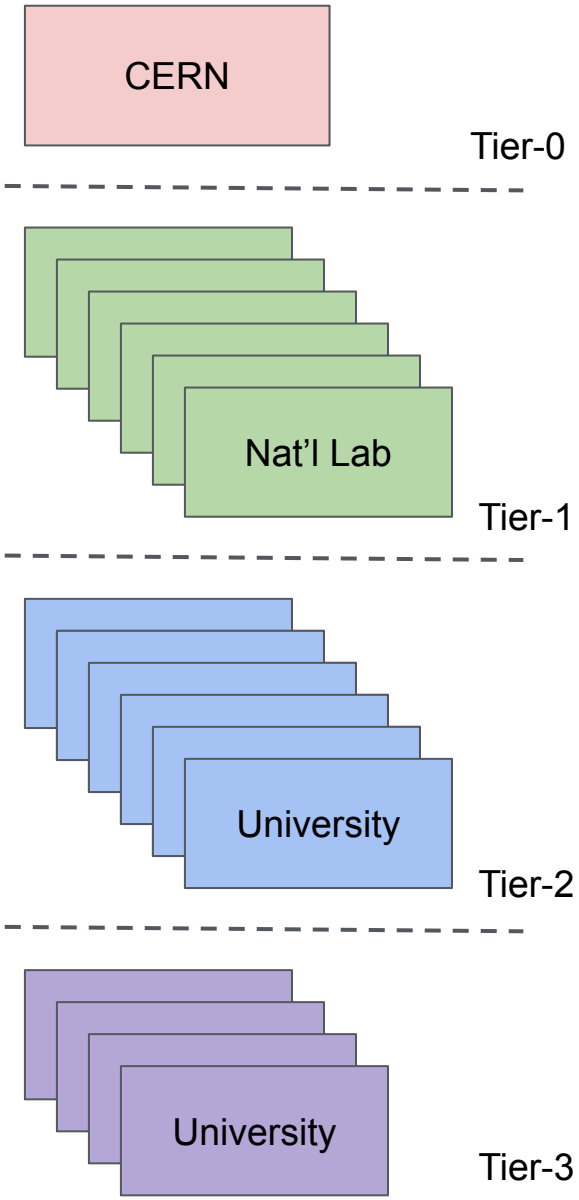
- The detector has 100s of millions of channels, and collisions happen 40 million times per second – In theory, this is 10^{17} bytes every second
- The trigger filters down these 40 million events/sec into a more manageable 2,000 events/sec – but even after this reduction, the amount of data remaining is still several gigabytes a second, approximately a DVDs worth of data for every second the detector runs
  - CMS has recorded 100,000,000 gigabytes of data, which is all securely archived on two separate tape backup systems
- Each collision can take up to a fill minute to process on a CPU. In total, we use 4.8 billion CPU-hours each year
- And this doesn't even consider all the simulated collisions we need to produce

CMS Computing Facilities

It's not feasible to have one large facility for all of CMS' computing needs. We instead have a large number of globally-distributed and interconnected facilities, called "The Grid". We divide the grid into four different tiers based on size:

- Tier-0: At CERN, has trigger and a tape copy of all RAW data produced by the detector
- Tier-1: At 7 national laboratories. The Tier-1s store an additional tape copy of all RAW data
- Tier-2: 60 large university facilities. Where most user analysis is run
- Tier-3: 20 Smaller facilities usually purpose-built by the institution. Fermilab is the exception with a very large Tier-3 for users

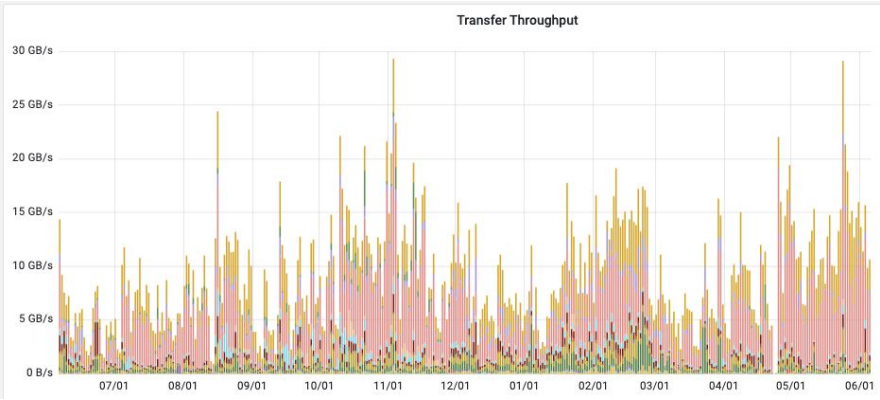
The sites are all connected by fast networking links, to allow the data to be quickly moved between sites



Moving Data Between Sites

With the sites being distributed globally, a key problem being able to move the data around globally.

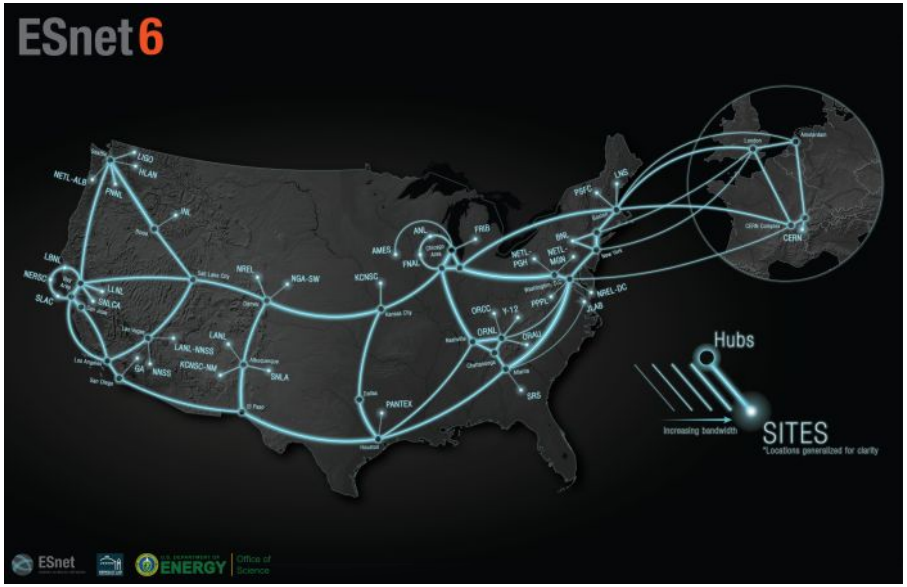
- Where possible, we “move computation to the data”



Within the USA, ESNet runs a private, Research and Education-only portion of the internet to connect all DOE labs and many Universities together

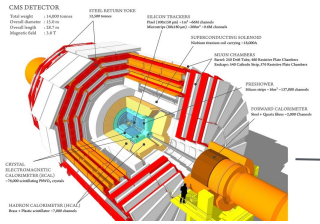
Even when the experiment isn’t running, CMS is moving several Gigabytes/sec across the network

Within the next decade, ESNet will have a Terabit/sec of bandwidth <u>Across the atlantic</u>



More Info: https://www.es.net/welcome-esnet6/

Preparing for Analysis: The Data Processing Workflow

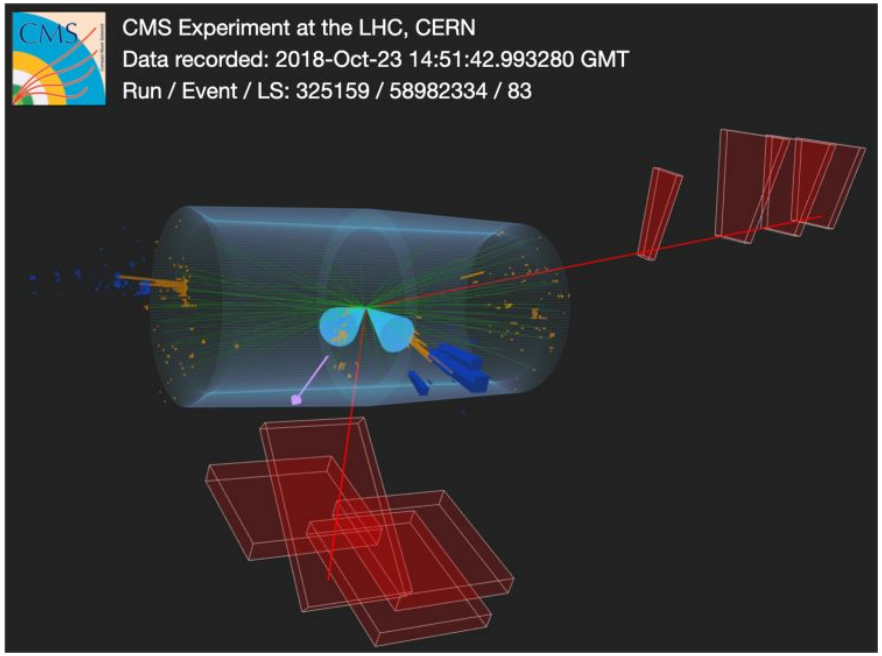


GEN-SIM-DIGI → Trigger → RAW → RECO → AOD → MINIAOD → NANOAOD

You are here!

The raw data (simulated or real) is not in a format very useful for analysis.

- The detector data is points in 3D-space where particles interacted w/sensors
- We want to study *particles*, so we have to do *reconstruction* to “connect the dots”



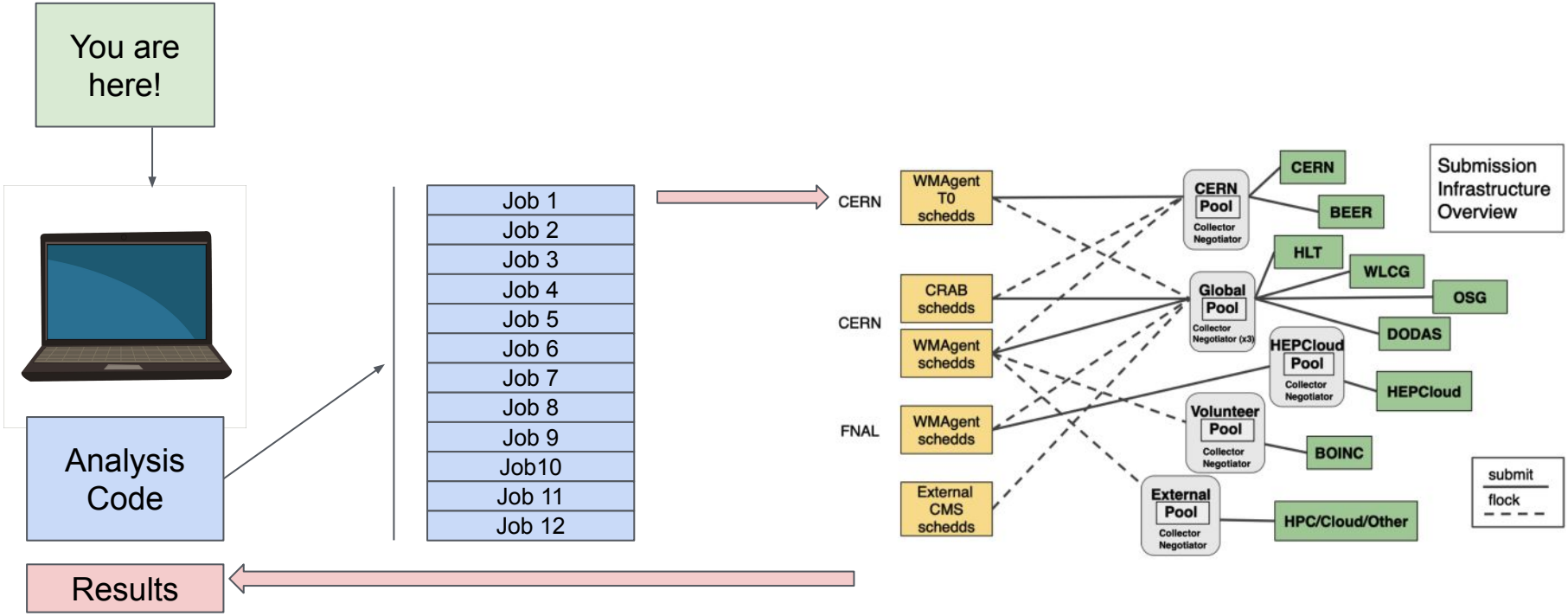
Visit https://tinyurl.com/cmsfnal23 for an interactive event browser

Processing Data at Scale

There is far too much data to reasonably process on a single computer. To process data at scale, we have to

- Split the data into smaller chunks
- Process each chunk on a separate machine
- Combine all the results together

To enable this, we developed The Global Pool, which combines all CMS' compute hardware into a single "virtual computer"

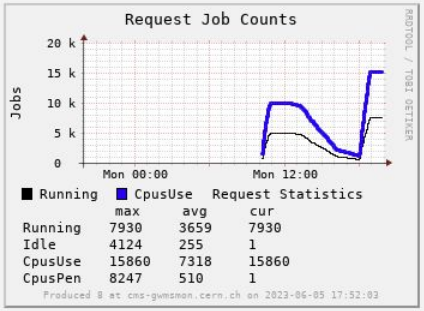
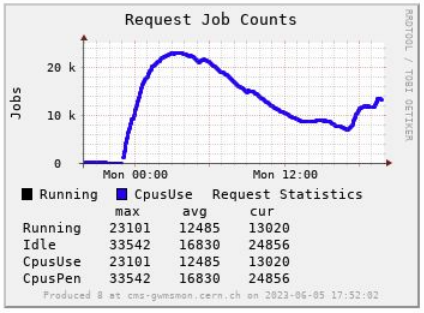


Img link

Processing Data at Scale

- Through the global pool, users can access 10s of thousands of CPUs
- HTCondor is used as our job scheduling system and is the entry point for all jobs to execute on CMS resources
- To be successful, you will become familiar with HTCondor

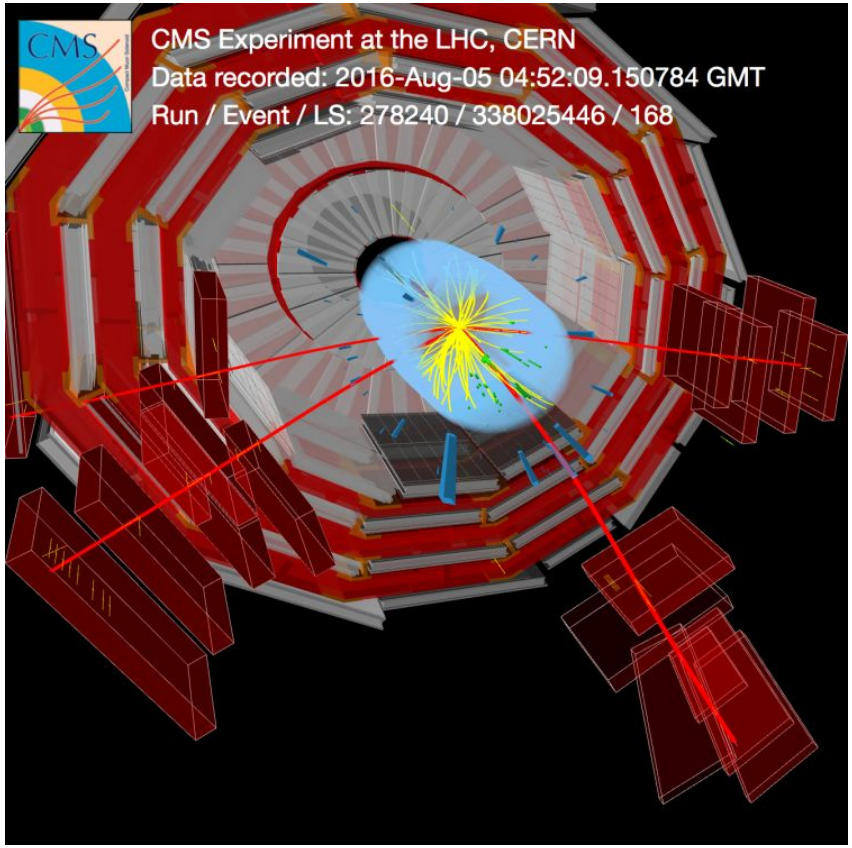
Idle Jobs/CpusPen	207995 / 396591
Unique Pressure	139623
Running Jobs/CpusInUse	51978 / 78985
Not Queued Jobs	103019
User Count	65
Tasks Running	1261
Tasks Held	20136
Update Time	6/5/2023, 12:51:46 PM

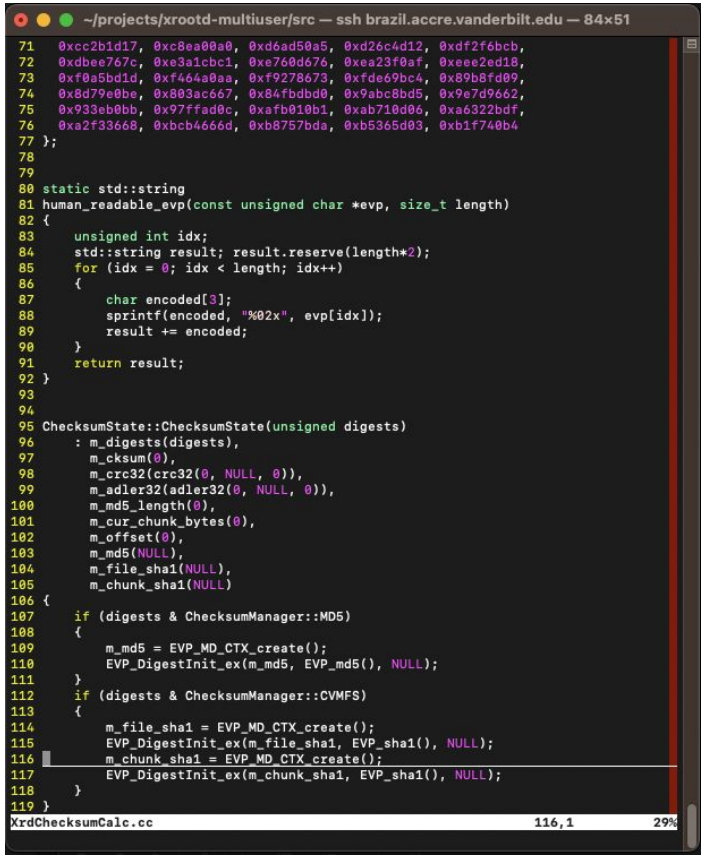
Username	Total	Running	Idle	CpusInUse	CpusPen	Tasks Running	Tasks Idle	Matching Site Count	User History	All ta
srosenzw	7,932	7,932	0	15,864	0	14	0	67		Held: Idle: Unex Subm Unkn Comp Remo Runn
iaandree	37,875	13,020	24,855	13,020	24,855	44	0	39		Held: Idle: Unex Subm Unkn Comp Remo Runn

Analyzing CMS Data

Expectation vs Reality

Many think that looking at CMS data is exactly that – visualizing events and drawing conclusions. There's way too many events to look at them manually. Instead we write software to examine the data for us and look at those results





X

✔

Analyzing CMS Data

To analyze CMS data one needs to

1. Access a CMS compute cluster via SSH
2. Develop and test code to process each event
3. Make HTCondor configuration files
4. Submit jobs to the grid

There will be more detailed talks/tutorials on each of these steps in the coming days!

Accessing Fermilab LPC Cluster via SSH

First step to using CMS' compute resources is to connect to them via SSH

SSH stands for "Secure SHell" and allows you to connect to a remote computer and type into it like you were physically there

CMS interactive machines come preconfigured with all the necessary software to write code and process data

1

2

```
~ — ssh lxplus.cern.ch — 80×25
┌─[meloam][melo-mbp][~]
└─▪ ssh lxplus.cern.ch
* ********************************************************************
* Welcome to lxplus711.cern.ch, CentOS Linux release 7.9.2009 (Core)
* Archive of news is available in /etc/motd-archive
* Reminder: you have agreed to the CERN
*   computing rules, in particular OC5. CERN implements
*   the measures necessary to ensure compliance.
*   https://cern.ch/ComputingRules
* Puppet environment: production, Roger state: production
* Foreman hostgroup: lxplus/nodes/login
* Availability zone: cern-geneva-a
* LXPLUS Public Login Service - http://lxplusdoc.web.cern.ch/
* An AlmaLinux8 based lxplus8.cern.ch is now available
* An AlmaLinux9 based lxplus9.cern.ch is now available
* Please read LXPLUS Privacy Notice in http://cern.ch/go/TpV7
* ********************************************************************
-bash: cite: command not found
-bash: cite: command not found
-bash: cite: command not found
┌─[meloam][lxplus711][~]
└─▪ hostname
lxplus711.cern.ch
```

https://uscms.org/uscms_at_work/computing/getstarted/uaf.shtml

Develop code to process each event

To process each event, a custom hand-written program with the analysis is run over all the data.

This code is unique to each analysis (though groups with similar analyses might share large parts)

Often, this analysis code is written in C++ which means that a typical day involves repeating:

- The code is written in a text editor (like VIM)
- The code is *compiled* into an executable
- The executable is then run over the data

```
void MyClass::Loop() {
  size_t nEvents;
  // load...

  for (Long64_t iEvent=0; iEvent<nEvents; iEvent++) {
    double MET_pt;
    int nElectron;
    double * Electron_pt;
    double * Electron_eta;
    // load...

    if ( MET_pt > 100. ) continue;

    for(size_t iEl=0; iEl<nElectron; ++iEl) {
      if ( Electron_pt[iEl] > 30. ) {
        hist->Fill(Electron_eta[iEl]);
      }
    }
  }
}
```

VIM Text Editor:
https://linuxconfig.org/vim-tutorial

Version Control with Git

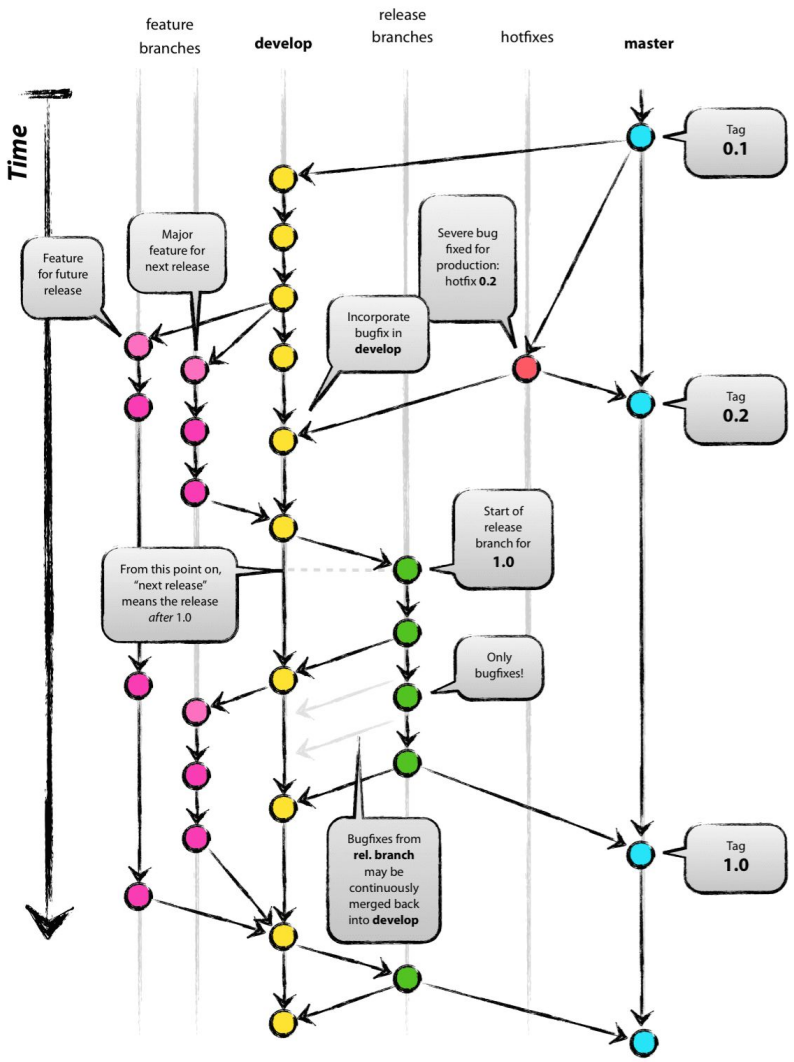
When developing an analysis, its important to:

- Collaborate with others
- Document the history of the code

Version Control enables both of these. Within CMS, we typically use the *Git* version control system

A typical Git workflow is:

- checkout the code from Git
- Write a new feature
- commit the code to Git
- push the code to a central location



See also: https://git-scm.com/video/what-is-version-control

Make HTCondor Config Files

With the code running functioning on a single machine, the next step is to write a configuration to tell HTCondor how to run our executable.

This configuration, called a JDL specifies:

- The executable to run
- Where to store the log files
- What files to move to/from the grid
- What arguments to pass to the executable
- How many copies of the job to run

HTCondor JDL

```
universe = vanilla
Executable = sleep.sh
should_transfer_files = YES
when_to_transfer_output = ON_EXIT
Output = sleep_$(Cluster)_$(Process).stdout
Error = sleep_$(Cluster)_$(Process).stderr
Log = sleep_$(Cluster)_$(Process).log
Arguments = 60
Queue 2
```

sleep.sh

```
#!/bin/bash
set -x
# Sleep
sleep $1
echo "##### HOST DETAILS #####"
echo "I slept for $1 seconds on:"
hostname
date
```

More Info:
https://uscms.org/uscms_at_work/computing/setup/batch_systems.shtml

Submit Jobs to the Grid

With the executable and the JDL, it's time to submit to the Grid.

```
[username@cmslpc132 ~]$ condor_submit sleep-condor.jdl
Querying the CMS LPC pool and trying to find an
available schedd...
Attempting to submit jobs to lpcschedd3.fnal.gov
Submitting job(s)..
2 job(s) submitted to cluster 76596545.
```

There are a number of HTCondor commands, but the most important two are:

- condor_submit
- condor_q

We <u>hope</u> that the Grid works 100% every time, but unfortunately there are several ways for the jobs to fail, so some "hand holding" has to be done to recover from failed jobs.

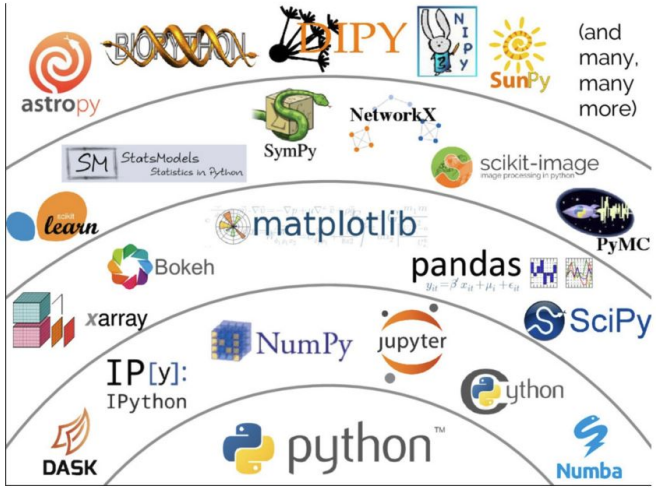
```
[username@cmslpc132 condor]$ condor_q

-- Schedd: lpcschedd3.fnal.gov : <131.225.188.235:9618?... @ 09/08/22 15:12:37
 ID          OWNER            SUBMITTED     RUN_TIME ST PRI SIZE CMD
76596545.0   tonjes           9/8  15:12   0+00:00:00 I  0    0.0 RunAN.sh 2018 MC

Total for query: 1 jobs; 0 completed, 0 removed, 1 idle, 0 running, 0 held, 0 suspended
Total for tonjes: 1 jobs; 0 completed, 0 removed, 1 idle, 0 running, 0 held, 0 suspended
Total for all users: 1208 jobs; 0 completed, 41 removed, 898 idle, 203 running, 66 held, 0 suspended
```

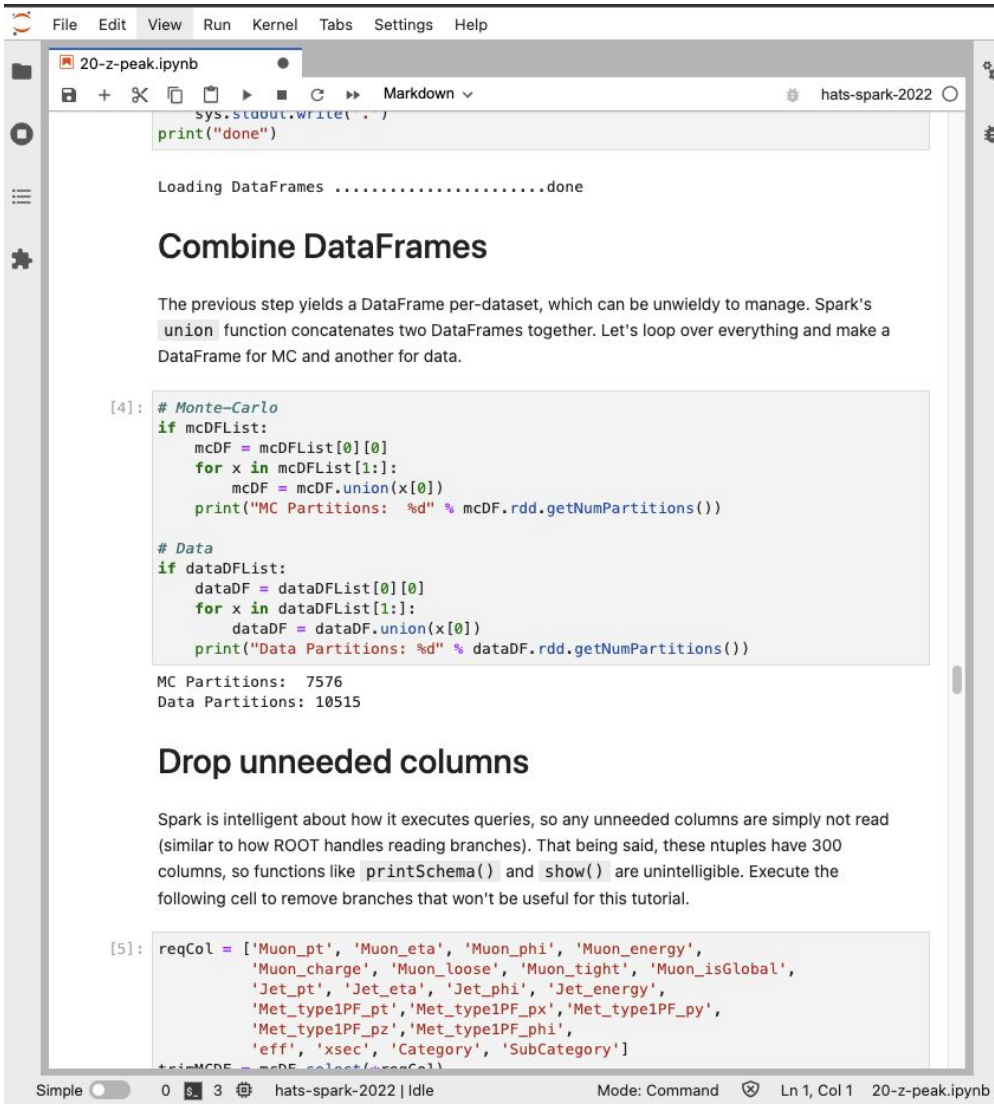
Notebooks and Python-based Analysis

Python Notebooks

In addition to the “traditional” analysis tools, CMS has begun to embrace the python-based Data Science tools embraced by industry.

Several of these “Analysis Facilities” are being developed, which use industry-standard tools like Pandas, Numpy and Dask





img

Python Notebooks

Python notebooks have several advantages:

- Plots and histograms can be displayed in-line with the code, making it obvious what code generated which plots
- The Python syntax can be much simpler to work with than the equivalent C++ code
- Because we're using libraries and packages from other fields, we take advantage of the significant effort others are putting into Data Science

The python code can also be used outside of notebooks, if one is accessing a more "traditional" cluster

Columnar Analysis

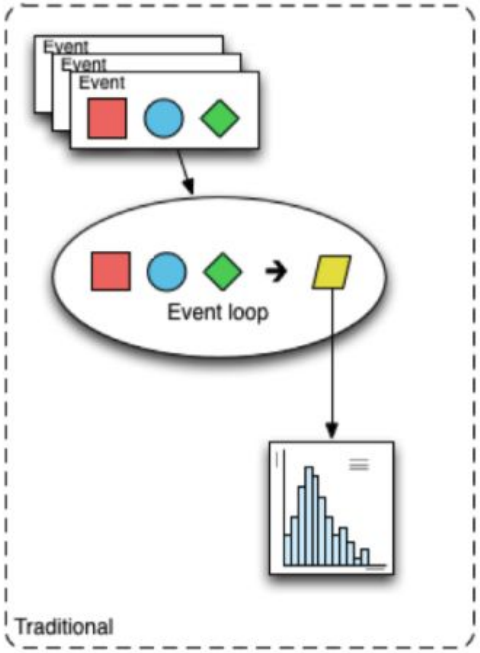
In a “traditional” analysis, each event is processed one-at-a-time, meaning:

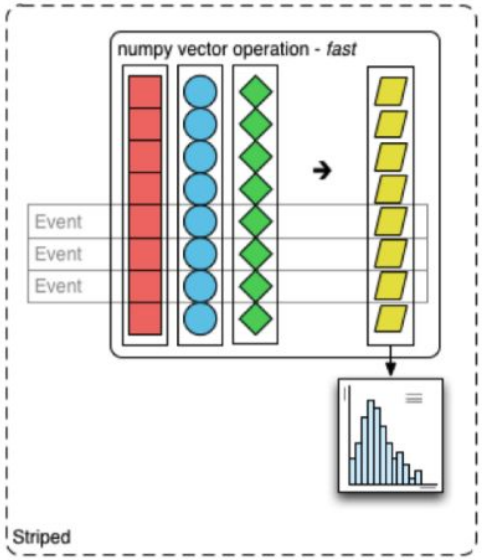
- An event is loaded and the appropriate values are extracted
- Some computation happens over this single event
- The temporary space is cleaned and the next event is loaded

In a “columnar” analysis, whole *batches* of events are processed at once, meaning

- 100s or even 1000s of events are loaded at once
- A bulk computation is done over the whole batch
- The temporary space is cleaned and the next event is loaded

The lower overhead for this method is particularly attractive for Data Scientists and is being embraced by CMS





GPUs and Accelerators

Why GPUs?

- We discussed rates and pileup before
- The time to process an event increases (roughly) by the square of the pileup
- During HL-LHC, the pileup will increase by a factor of two
- This means (roughly), the time to process each event increases by a factor of four!!
- Due to other affects, the worst-case estimate for HL-LHC is a need for 10x the amount of CPUs
- Will be very hard to make this work financially, need to speed things up!

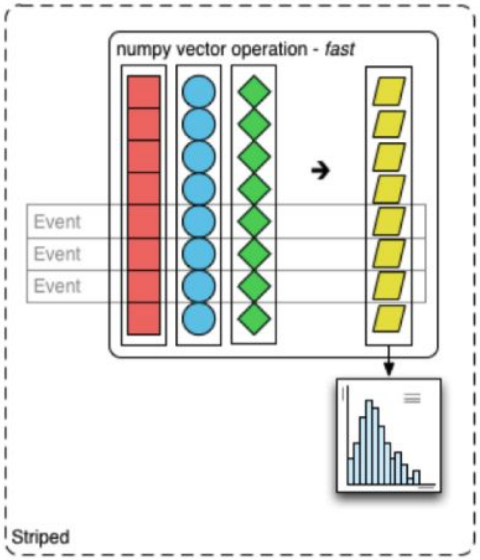
GPUs

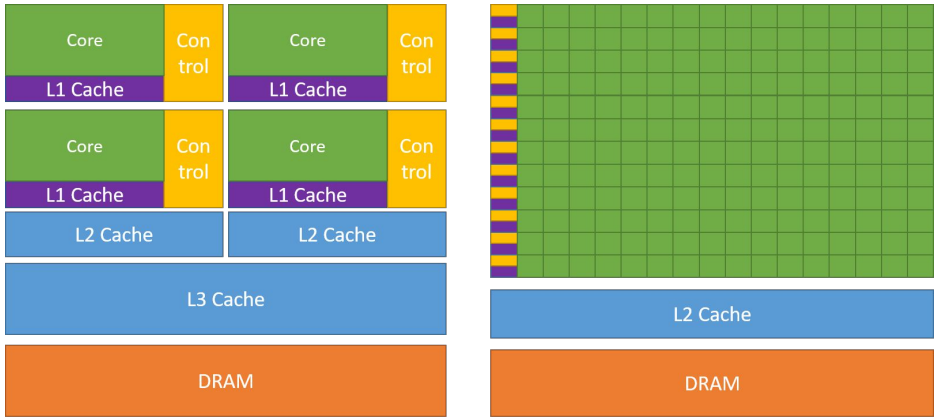
One path CMS is exploring to accelerate our workflows is to exploit GPUs

GPUs are much faster both per-dollar and per-watt of performance, however you can't simply take code written for CPUs and run them on a GPU, you need to rewrite them

Several people are targeting the most-time consuming portions of our workflow to port to GPUs, and they have reached 30% of the modules, with more improvements to come

Accelerator	Architecture	Socket	FP32 CUDA Cores	FP64 Cores (excl. Tensor)	M INT: C
H100	Hopper	SXM5	16896	4608	168
A100 80GB	Ampere	SXM4	6912	3456	691
A100 40GB	Ampere	SXM4	6912	3456	691





CPU GPU

img

GPUs for Machine Learning (ML)

Recently, the use of GPUs for AI/ML has exploded

- ChatGPT, etc are in the news constantly

There are a number of efforts within CMS to use Machine Learning (ML) to improve all phases of data processing, including:

- Anomaly detection for detector performance
- “Tagging” physics objects in the detector
- Full end-to-end ML-based reconstruction

There are a lot of opportunities to exploit ML within CMS, but a lot of questions remain from how to support them at the sites to how do we validate the results that come out of these algorithms.

The Future is Bright!

…come join us!

Questions?