HIGH ENERGY PHYSICS ON THE OSG

Brian Bockelman

CCL Workshop, 2016

SOME HIGH ENERGY PHYSICS ON THE OSG (AND OTHER PLACES TOO)

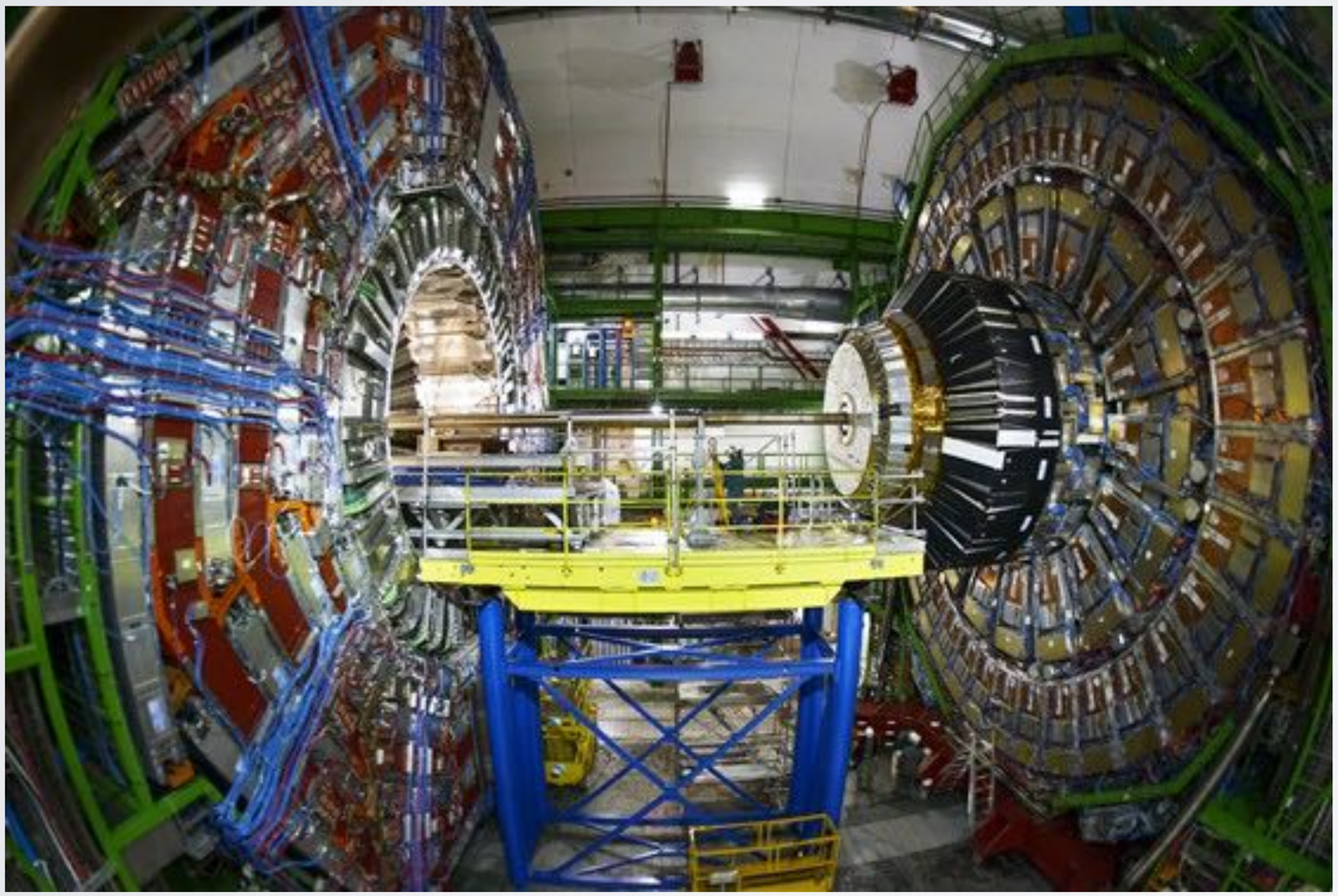
Brian Bockelman

CCL Workshop, 2016

Remind me again -

WHY DO PHYSICISTS NEED COMPUTERS?

WHERE DO EVENTS COME FROM?



- The obvious source is the detector itself.
- We must take the raw readouts and reconstruct them into physics objects.
- These objects represent things that have meaning to a physicist (muons, electrons, jets of particles).

LIFETIME OF A SIMULATED EVENT

- **GEN** - Given a desired physics signal, generate a particle decay from the random number generator.

- **SIM** - Given the GEN output, simulate the particles' paths and decay chains.

- **DIGI** - Given the simulated particles, simulate the detector readout.

- **RECO** - Reconstruct detector readout into physics objects.

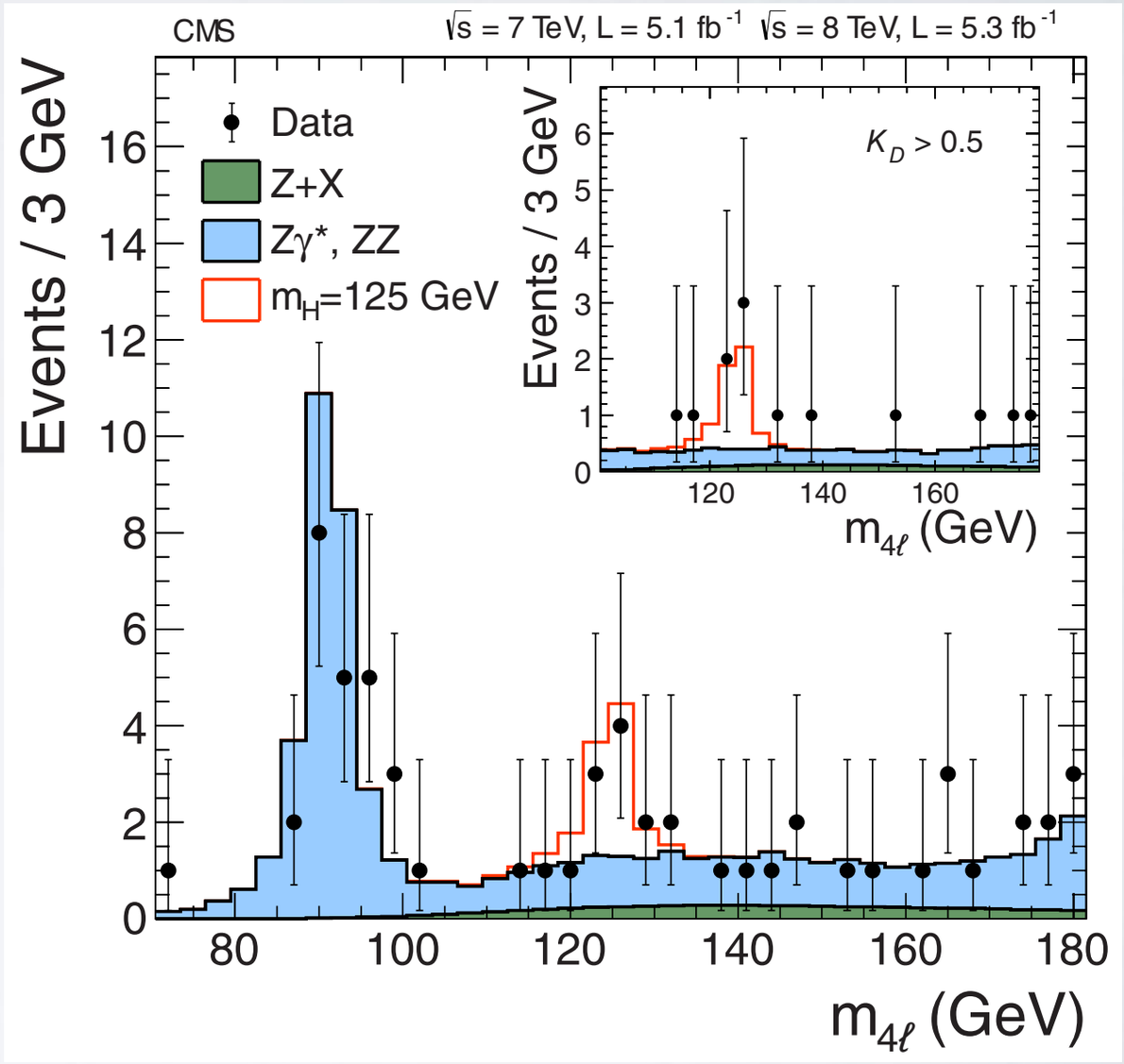
SIMPLE STATS FOR THE LHC AND CMS 2016 EDITION

- 40MHz of "bunch crossings"; each crossing results in about 25 particle collisions. **One billion collisions a second**.
    - Most are "boring" (for some definition of boring). We write out **1,300 events / seconds to disk**.
- 85 days of running time / year = 10B recorded events.
- For CMS, reconstruction takes about 14s / event. Reconstruction of the year's dataset is **54,000 CPU-months**.
- We aim for 1.3 simulated events per "real" event. GEN-SIM takes 44s / event and DIGI-RECO takes 26s / event: **350,000 CPU-months**.
- CPU requirements go up quadratically with number of collisions per beam crossing. We expect an increase from 25 to **35 next year**.
- Depending on the data format used, the event size is 30KB to 500KB.

(Note: all numbers given are correct to the order-of-magnitude; accurate current performance information is considered private)

AND FINALLY, ANALYSIS

- After reconstruction of data and simulated events, we deliver groups of events into coherent *datasets* to physicists.
- The physicists scan the datasets, comparing the number of recorded events with a given signature against the expected number from known physics.
  - A discovery requires 5-sigma deviation of the signal from the expected behavior.
  - Determining these uncertainties is what drives the need for simulation.
- CPU and IO needs of analysis vary by two orders of magnitude - depends on the physicists.
  - Needs are difficult to model! I think of it as a fixed percentage (60%) of centralized production needs.



HOW DO WE DO IT?

DISTRIBUTED HIGH THROUGHPUT COMPUTING

- Practically every HEP experiment has built their computing infrastructure around the concept of **distributed high throughput computing (DHTC)**.

- High-Throughput Computing: maximizing the usage of a computing resource over a long period of time. "FLOPY, not FLOPS".

- Distributed HTC: Utilizing a variety of independent computing resources to achieving computing goals. "The Grid".

THE OPEN SCIENCE GRID

- The OSG is a "national, distributed computing partnership for data-intensive research".
    - Consists of a fabric of services, software, and a knowledge base for DHTC.
    - Partnership is between different organizations (science experiments, resource providers) with an emphasis on *sharing of opportunistic resources* and enabling DHTC.
- Around 50 different resource providers and 170k aggregate cores.

In the last 24 Hours	
403,000	**Jobs**
1,423,000	**CPU Hours**
6,691,000	**Transfers**
900	**TB Transfers**
In the last 30 Days	
10,268,000	**Jobs**
118,604,000	**CPU Hours**
81,626,000	**Transfers**
13,431	**TB Transfers**
In the last 12 Months	
153,589,000	**Jobs**
1,250,239,000	**CPU Hours**
1,855,862,000	**Transfers**
218,000	**TB Transfers**

FIRST, YOU NEED A POOL

- One of the most valuable services OSG provides is a **HTCondor-pool-on-demand**.
  - You provide the HTCondor submitters (`condor_schedd`) and a credential; we provide HTCondor worker nodes (`condor_startd`) from various OSG resources.
  - Bulk of these worker nodes come from the OSG *Factory* submitting jobs to a remote batch system through a *Compute Element*. These *pilot jobs* will be started by the site batch system and launch the `condor_startd` process.
    - Don't think of this as submitting jobs to a batch system, but rather as a ***resource acquisition***.
    - Resources might be ones you own, opportunistic resources, or some combination.
- Allows the experiment to view the complex, heterogeneous grid as a single pool of resources.
- Not all organizations will use the OSG-provided factory and interact directly with the CE; all currently use the same pilot model. Other important examples include **PanDA** and **DIRAC**.

WORKFLOWS

- Once we have a pool of compute resources, we divide the work into a series of workflows.

- Typically, each workflow works on an input dataset, requires some physics configuration file, and has an output dataset.

- Workflows are often grouped into "campaigns". "Process all 2016 detector data using CMSSW_8_0_20 with the new conditions."

WORKFLOWS

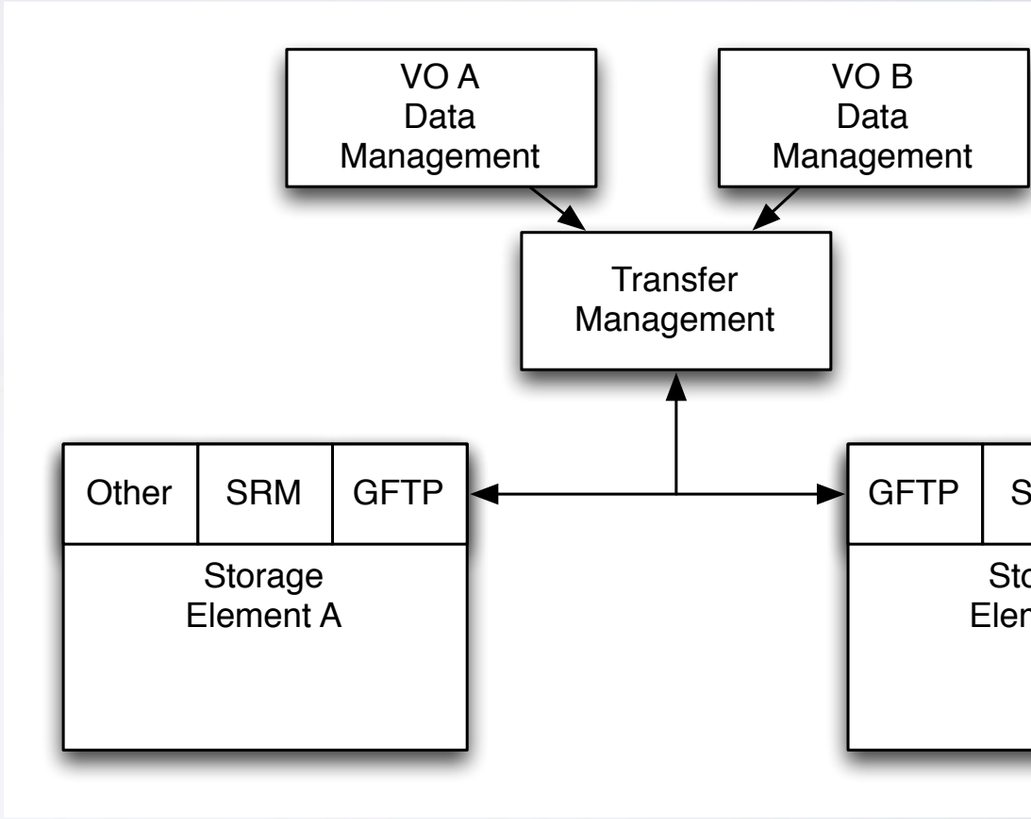
- Processing a dataset requires the workflow broken down into a series of jobs. I.e., Job XYZ will process events 1000-2000.
    - When the job is materialized - and whether it is static or dynamic - greatly differs by experiment.
- Often, there are only loose dependencies between jobs (if any at all). Dependencies are often not staticky defined: a "merge job" may be created once there is 2GB of unmerged output available.
- I can think of only one example (Lobster) where a non-HEP-specific workflow manager was used for a HEP workflow.

PORTABILITY

- Once upon a time, the LHC experiments could only run jobs at LHC sites: LHC jobs needed LHC-specific services, LHC-specific storage systems, and extremely-large, finicky software stacks.
    - This implied LHC-specific sysadmins! You don't want to be the site paying $100k/yr to the sysadmin for $50k of hardware.
- Over the past 3-5 years, great strides were made to simplify operations:
    - **CVMFS** (discussed elsewhere) provides a mechanism to easily distribute software.
    - LHC-specific features were removed from storage systems. Currently, we can run on top of a generic **shared POSIX-like filesystem**.
    - The event data was made more portable with remote streaming (more later).
    - LHC-specific data services were either eliminated, centralized, or made generic (i.e., **HTTP proxy server**).
- Today, our site requirements are basically **RHEL6, robust outgoing network connection, HTTP proxy, and CVMFS**.

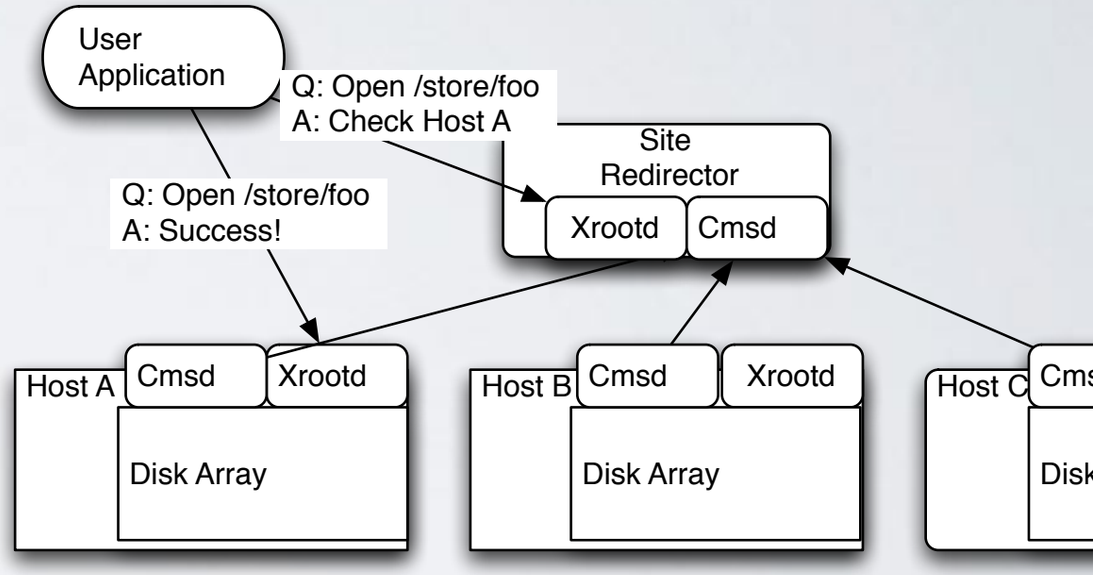
DATA MANAGEMENT

- HEP experiments have a huge bookkeeping problem:
  - A dataset is a logical group of events, typically defined by their physics content. Commonly stored as files in a filesystem.
  - We have thousands of new datasets per year, each with 10's to 10,000's of files.
    - CMS manages O(50PB) of disk space across O(50) sites.
- Most experiments develop a **bookkeeping** system to define the file<->dataset mapping and hold metadata; a **location service** to determine where files are currently placed; and a **placement service** to determine what movement needs to occur.
  - Surprisingly, most use a common **transfer service** (FTS) to execute the decisions of the placement service.
  - The past is littered with the bodies of "generic" bookkeeping, location, and placement services: it seems the requirements depend heavily on the experiment's computing model.



DATA PORTABILITY

- About 5 years ago, the only way to read a single event was submit a job to the datacenter holding the file (and wait in line!).
- We have been heavily investing into remote streaming from storage to the end-application.
  - Using "data federations" to hide many storage services behind a single endpoint.
  - Altering the application to be less sensitive to latency.
- Originally, used for preventing application failures and user usability improvement.
  - It's become critical for previously-impossible use cases.
  - Allows for processing-only sites.

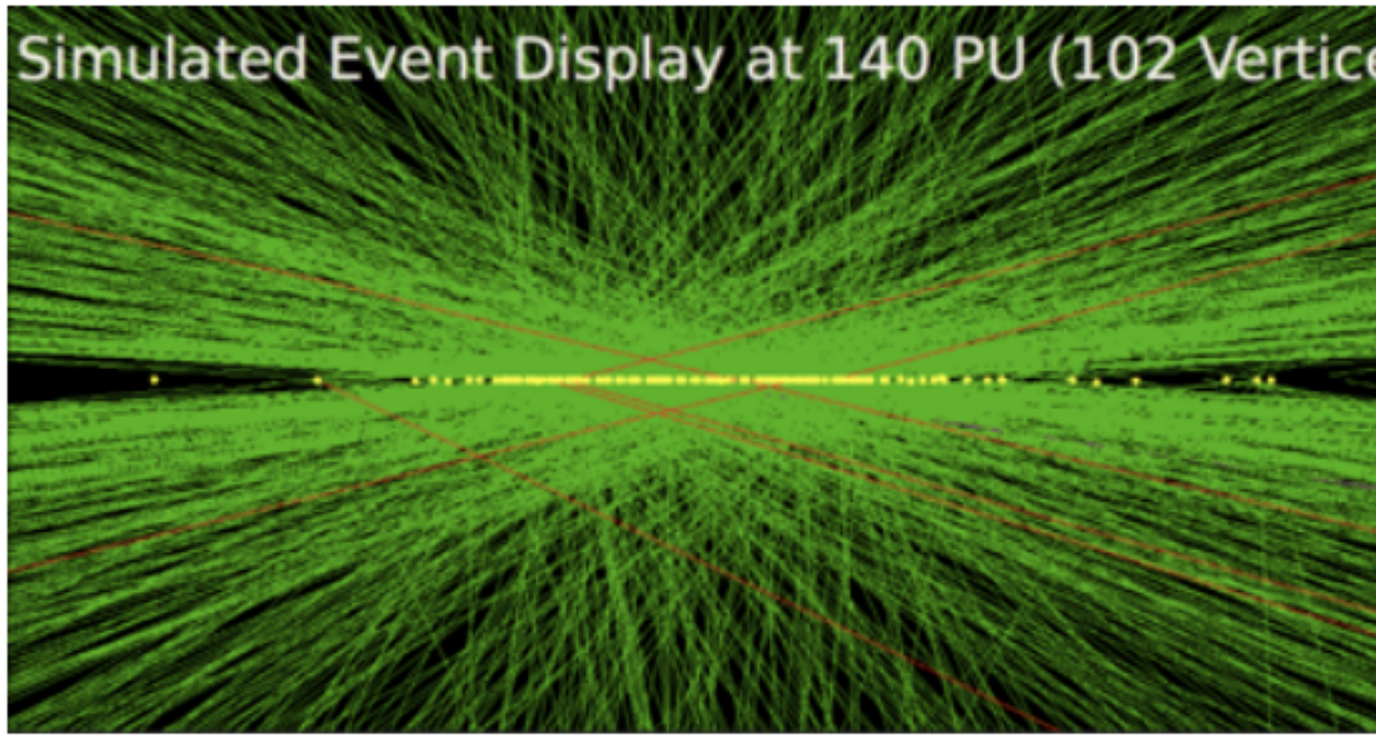


CHALLENGES FOR HEP

FASTER, BETTER, CHEAPER

(Pick Three)

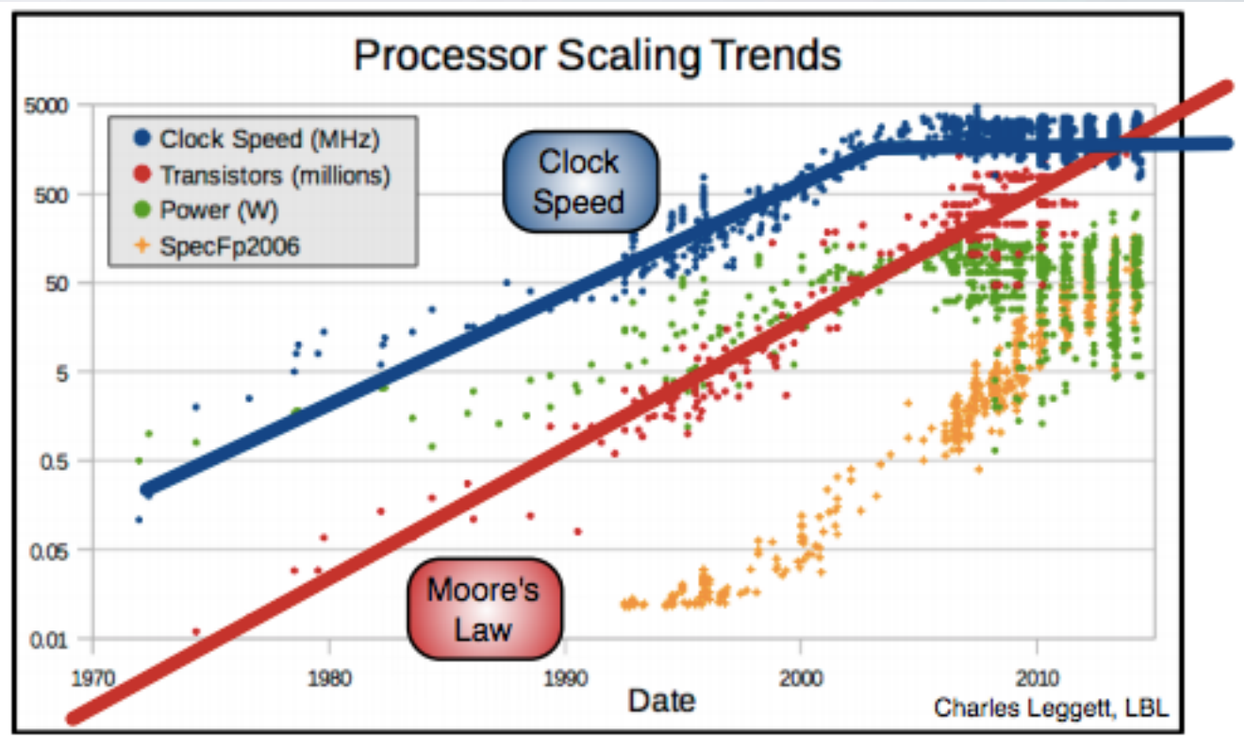
- In the short term, the LHC is taking much more data than expected.
- In the long term (10 years), the LHC's CPU requirements are 60x today's.
- Moore's Law will likely take care of the first 10x.
    - Prognosis for a 6x budget increase … not good.

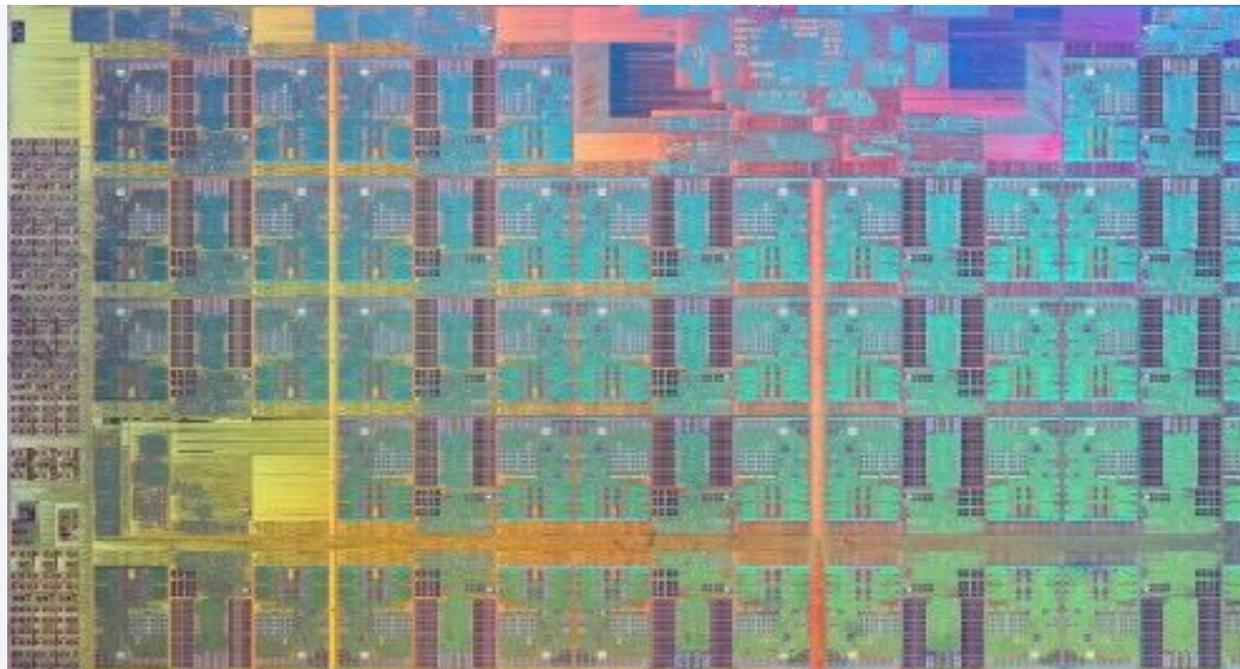


CMS Simulation
Erica Brondolin, HEPHY

THE RETURN OF HETEROGENEITY

- In the Bad Old Days, there was practically a different processor architecture for each cluster.
- This may occur again if GPUs and PowerPC or ARM become *way* more popular.
    - More likely: the base ISA is x86, but performance differs by 4x depending on available extensions.
- What workflow, compiler, and software design issues occur when you have to tune for both Intel Atom and KNL?





THE OTHER KIND OF HETEROGENEITY

- The Grid has a huge variation in hardware but a strong source of homogeneity: admins paid to do HEP and a belief in DHTC.
- To meet future resource needs, working to incorporate other economic models (besides buying clusters):
  - *Cloud resources*: infinite-ish elasticity but high-cost.
  - *Supercomputer allocations*: "cheaper", lots of CPU for peak processing. Optimized for a different IO profile, inflexible environments, organizationally not used to serving a global community.
- **Technical issues abound**! How do you integrate 100,000 cores with relatively poor network connectivity and no CVMFS? How do you work with an organization with little history in automating workflows?
  - **Non-technical issues abound**: experiments plan at the 2-10 year timescale. Allocations occur yearly.

DID I MENTION PORTABILITY?

- As the LHC experiments get closer to resource starvation (hungrier!), there is an increased interest in portability:
    - OSG is investigating adoption of Singularity; would allow experiments to "pick their OS" environment.
    - CVMFS is taking several parallel strategies to function at HPC sites.
    - Caching is an increasingly common strategy for data access for latency sensitive workflows - and continued work to make more workflows latency-friendly (enables remote streaming).
    - Access site computing resources via "just SSH": no CE needed.
- **What else can we do to lower the bar for sites?**

JOBS ARE NOT THE BEST ABSTRACTION

- How many events should go into a job?
    - There is likely 100x difference in processing power between our largest HTCondor worker nodes (Intel KNL with 256 threads) and smallest (BOINC volunteer laptop). A 2-hour job (too short) on the KNL is more than a week (too long) on the laptop.
        - We may not know how much time we have on the resource: little-or-no warning before preemption.
        - We don't always have a reliable estimate of job length.
        - All of the above issues get **worse every year**.
- Several community projects (notably, PanDA's **JEDI** and ATLAS's EventService) to allow jobs be dynamically defined and change during runtime.
    - Eventually, becomes an **immense bookkeeping exercise**.

STORAGE IS EXPENSIVE

Running it is even more expensive

- The "storage element paradigm" results in a mess. We assume (incorrectly):
  - The experiment can keep track of the data it wrote to the SE.
  - The SE is available and functioning. The sysadmin responds to tickets and understands what the VO is asking.
  - Files, once written, are neither deleted nor corrupted.
- Not to mention, running large robust shared filesystems remain a dark art! This motivates a lot of **interest in caching** in our field.
- The SE is overkill for most of our workflows. Setting aside user interaction, the semantics needed are much closer to an object store. Could we take advantage of this?

PARTING SHOTS

- My goal today was to share (a) why HEP has significant computing needs, (b) how we currently solve the problem, and (c) some outstanding problems you might be interested in tackling.

- HEP is a field with a long history of computing. This is a curse and a blessing as we look to our next set of scaling challenges.

- I think both sides benefit when HEP adopts outside technologies rather than "build our own." See HTCondor as a great example.

QUESTIONS?
COMMENTS?
HECKLING?