Meeting Future Software Challenges in High-Energy Physics

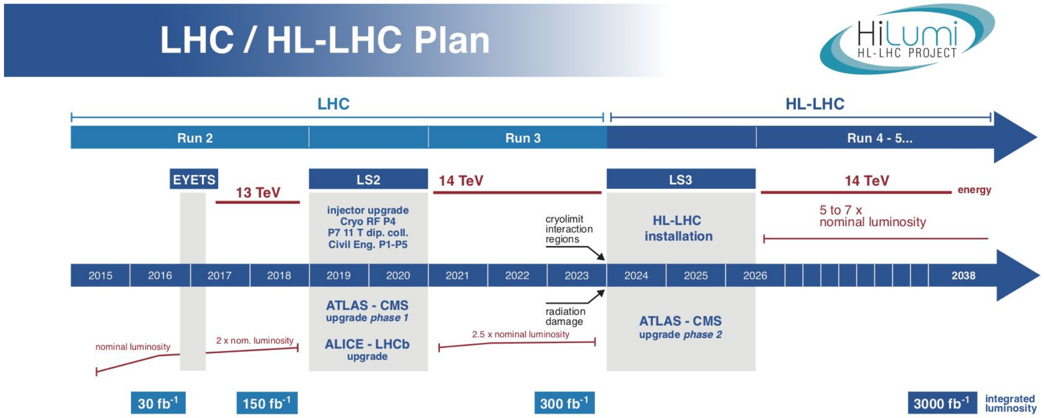
Graeme A Stewart, CERN EP-SFT

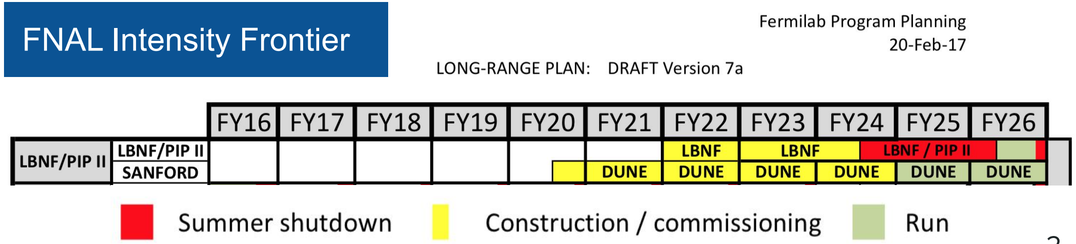
University of Geneva, 22 May 2019

HL-LHC and the Intensity Frontier

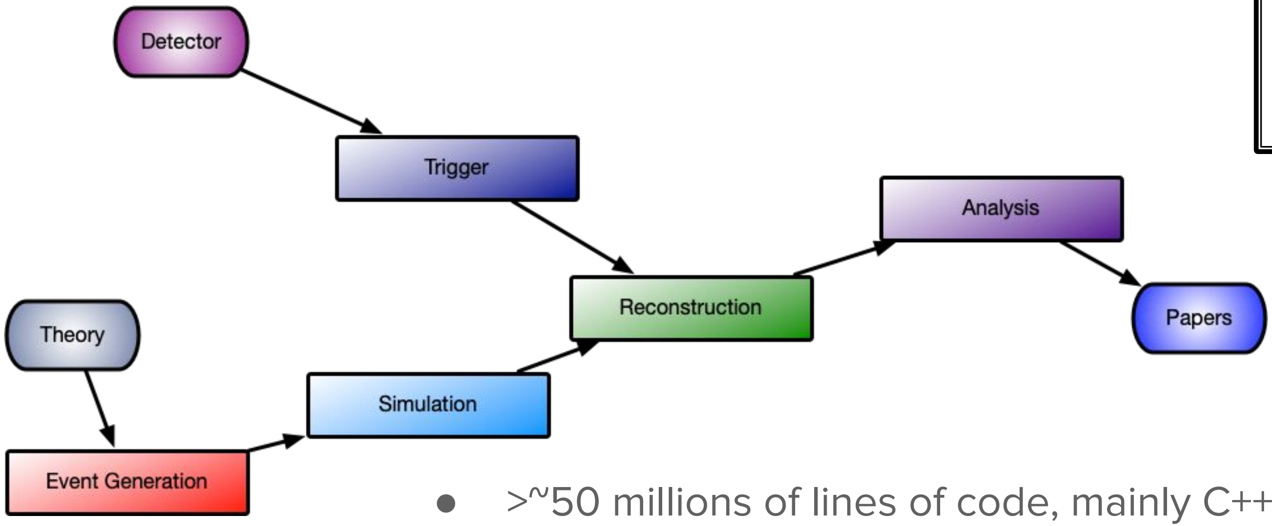
Our mission:

- Exploit the Higgs for SM and BSM physics
- b, c, tau physics to study BSM and matter/anti-matter
- Dark matter
- QGP in heavy ion collisions
- Neutrino oscillations and mass
- Explore the unknown





An Overview of HEP Software

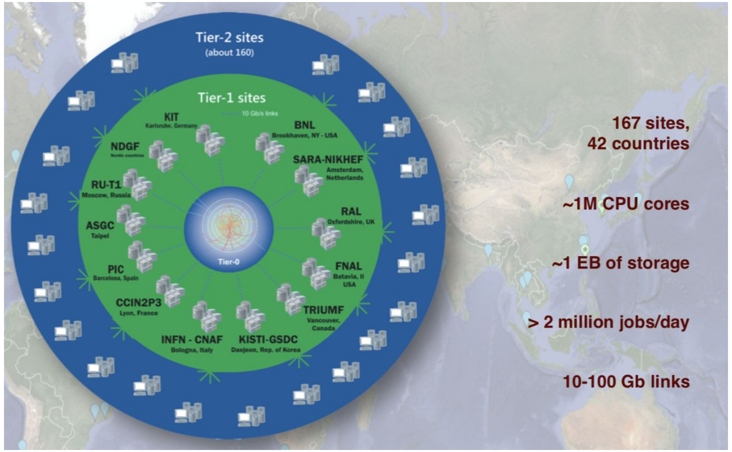


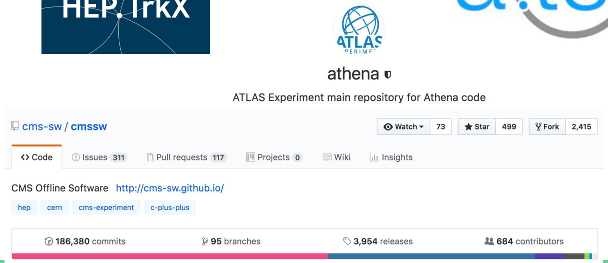
This is the “traditional” view and **how this changes** in the future is an important topic for our discussions

- >~50 millions of lines of code, mainly C++, a lot of Python
  - Commercial development cost ~500M CHF
- Critical part of our physics production pipeline, from triggering all the way to analysis and final plots as well as simulation
- Significant pieces of software are already shared by most experiments:
  - Event generators, Geant4, ROOT

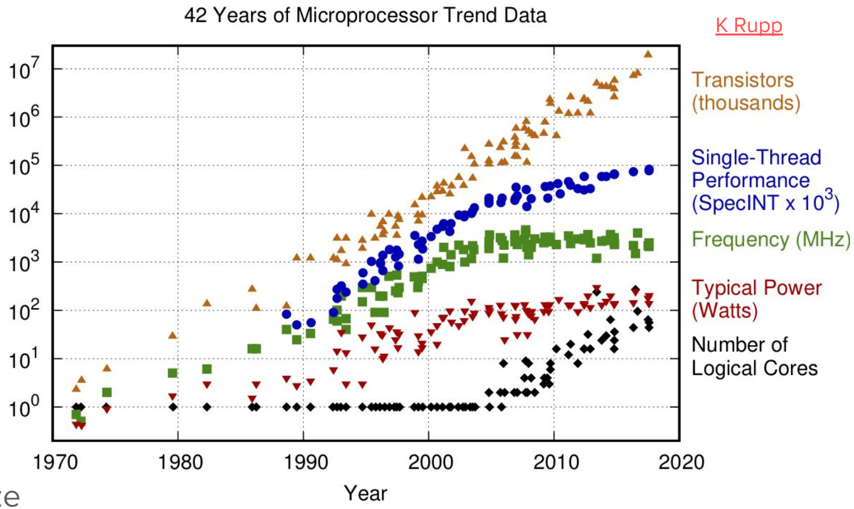
HEP Computing

- Tasks broken into jobs by experiment production systems (levels of parallelism)
  - Tasks ➔ job ➔ events ➔ algorithms
- LHC experiments use
  - 1M CPU cores every hour of every day
  - Store 1000PB of data (600/400PB tape/disk split)
    - We are in the exabyte era already
  - 100PB of data transfers per year (10-100Gb links)
- This is a huge and ongoing cost in hardware and human effort
- With significant challenges ahead of us to support our ongoing physics programme

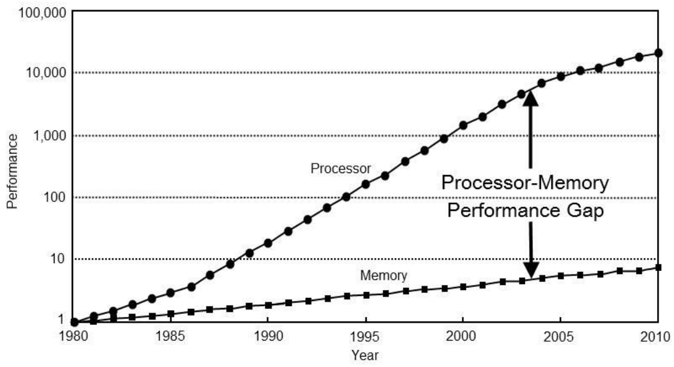




Technology Evolution



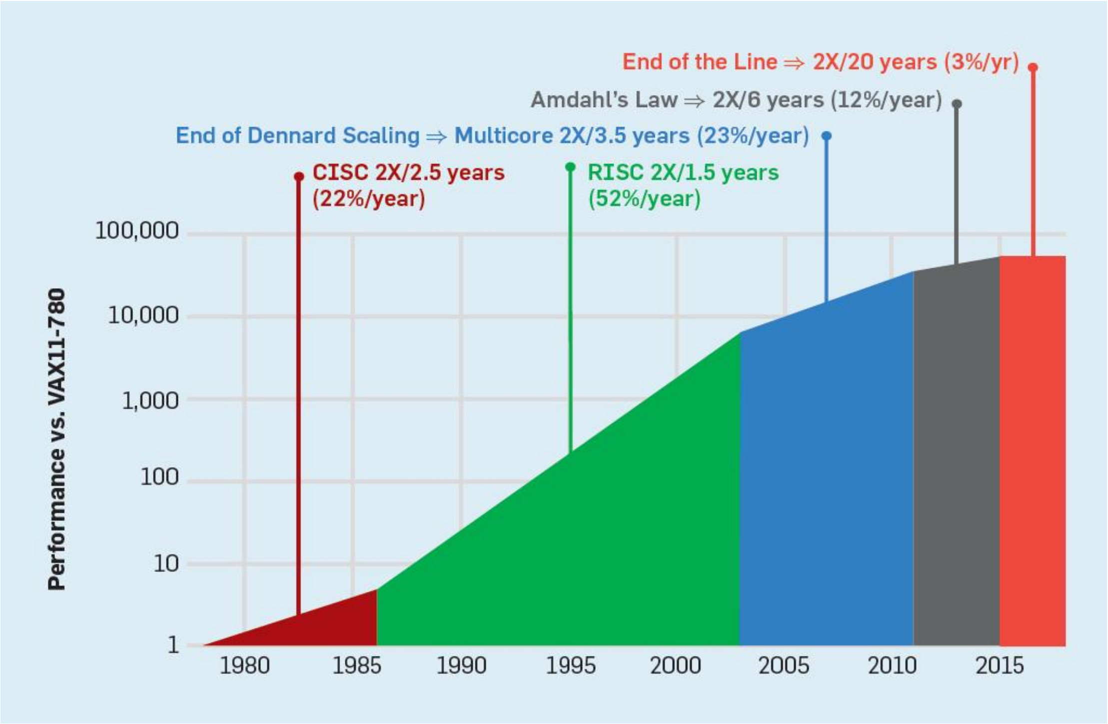
- Moore’s Law continues to deliver increases in transistor density
  - But, doubling time is lengthening
- Clock speed scaling failed around 2006
  - No longer possible to ramp the clock speed as process size shrinks
  - Leak currents become important source of power consumption
- So we are basically stuck at ~3GHz clocks from the underlying Wm^{-2} limit
  - This is the *Power Wall*
  - Limits the capabilities of serial processing
- Memory access times are now ~100s of clock cycles
  - Poor data layouts are catastrophic for software performance



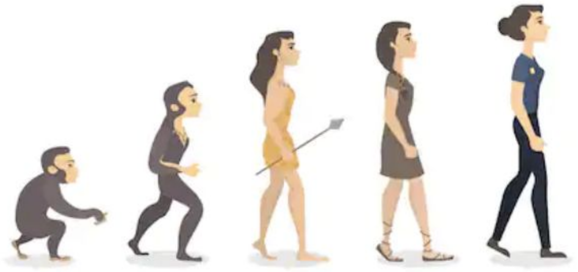
Decreasing Returns over Time

- Conclusion is that diversity of new architectures will only grow
- Best known example is of GPUs

[link]

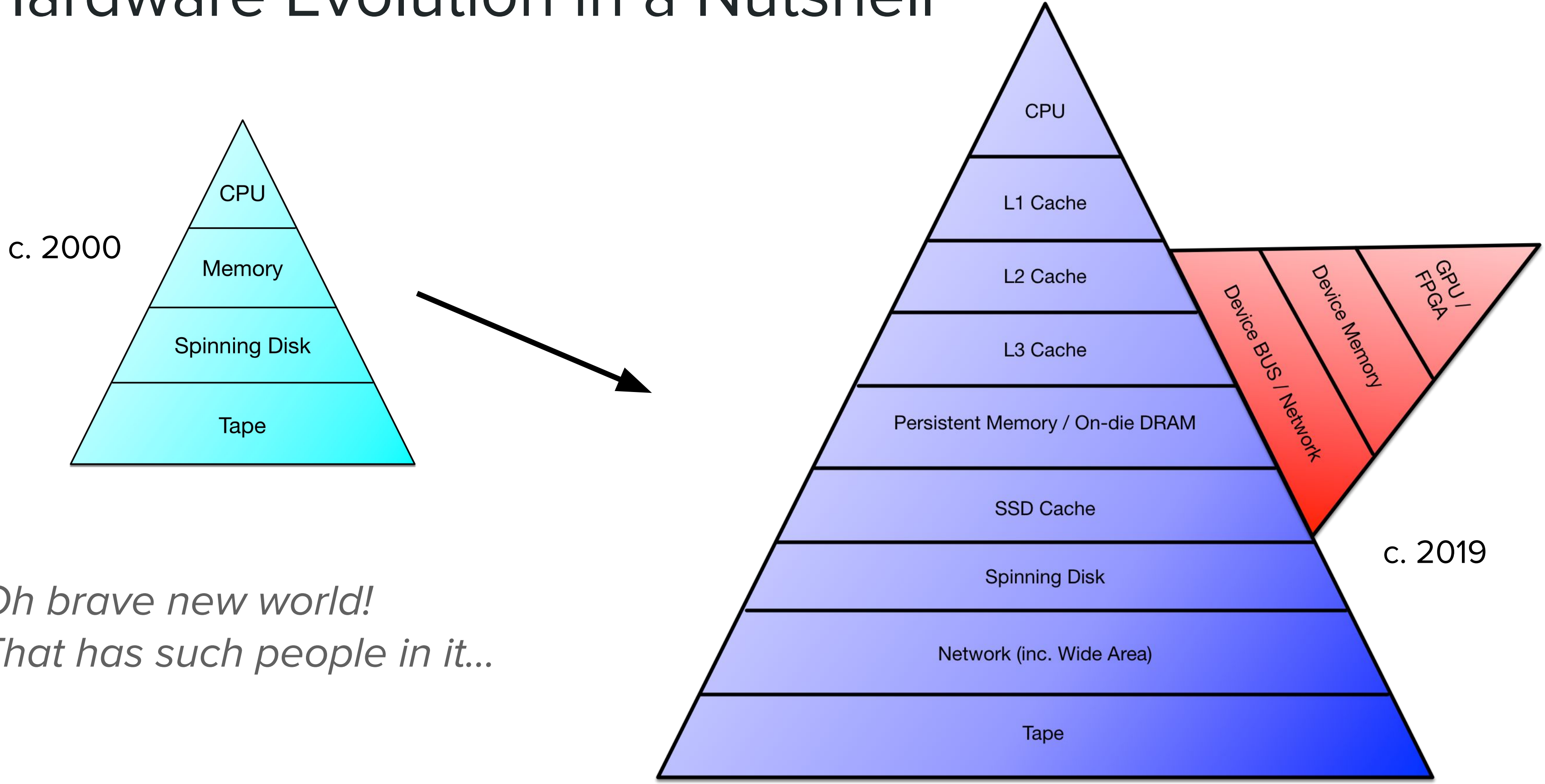


Drivers of Technology Evolution



- Low power devices
  - Driven by mobile technology and Internet of Things
- Data centre processing
  - Extremely large clusters running fairly specialist applications
- Machine learning
  - New silicon devices specialised for training machine learning algorithms, particularly low precision calculations
- Exascale computing
  - Not in itself general purpose, but poses many technical problems whose solutions can be general - HEP pushed to use HPC centres, especially in US
- Energy efficiency is a driver for all of these developments
  - Specialist processors would be designed for very specific tasks
  - Chips would be unable to power all transistors at once: dark silicon is unlit when not used

Hardware Evolution in a Nutshell



Oh brave new world!
That has such people in it…

Software Challenges and Opportunities

Concurrency

- The one overriding characteristic of modern processor hardware is concurrency
  - SIMD - Single Instruction Multiple Data (a.k.a. vectorisation)
    - Doing exactly the same operation on multiple data objects
  - MIMD - Multiple Instruction Multiple Data (a.k.a. multi-theading or multi-processing)
    - Performing different operations on different data objects, but at the same time
- Because of the inherently parallel nature of HEP processing a lot of concurrency can be exploited at rough granularity
  - Run many jobs from the same task in parallel
  - Run different events from the same job in parallel
- However, the push to highly parallel processing (1000s of GPU cores) requires **parallel algorithms**
  - This often requires completely rethinking problems that had sequential solutions previously, e.g. finding track seeds via cellular automata (TrickTrack library, CMS and FCC)

Heterogeneity

- There are a lot of possible parallel architectures on the market
  - CPUs with multiple cores and wide registers
    - SSE4.2, AVX, AVX2, AVX512, Neon, SVE, Altivec/VMX, VSX
  - GPUs with many cores; FPGAs
    - Nvidia (many generations - often significantly different), AMD, Intel, ...
- In addition there are ‘far out’ architectures proposed, like Intel’s Configurable Spatial Architecture
- Many options for coding, both generic and specific:
  - Cuda, TBB, OpenACC, OpenMP, OpenCL (➔ Vulcan), alpaka, Kokkos, ...
- Frustratingly no clear winner, mutually exclusive solutions and many niches
  - One option for now is to isolate the algorithmic code from a ‘wrapper’ that targets a particular device or architecture - approach of ALICE for their GPU/CPU code
  - Hiding details in a lower level library (e.g. VecCore) also helps insulate developers

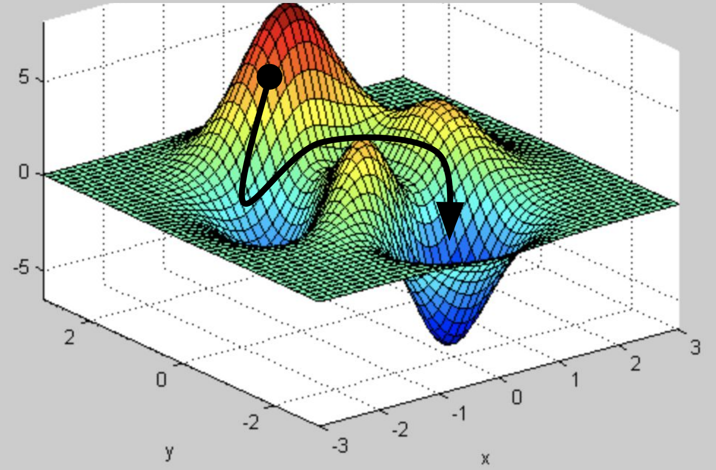
Data Layout and Throughput

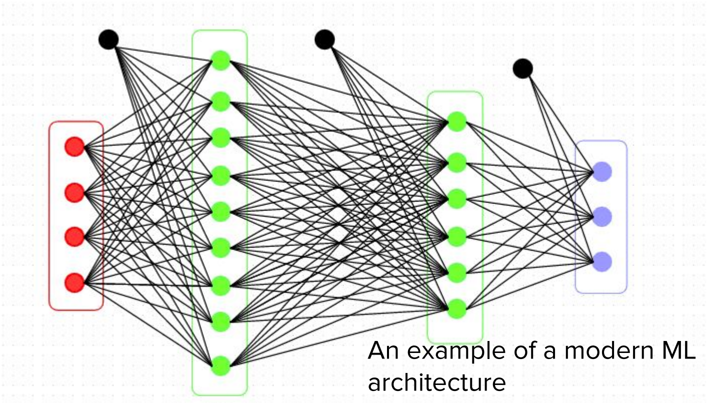
- Original HEP C++ Event Data Models were heavily inspired by the Object Oriented paradigm
    - Deep levels of inheritance
    - Access to data through various indirections
    - Scattered objects in memory
- Lacklustre performance was ~hidden by the CPU and we survived LHC start
- In-memory data layout has been improved since then (e.g. ATLAS xAOD)
    - But still hard for the compiler to really figure out what's going on
    - Function calls non-optimal
    - Extensive use of 'internal' EDMs in particular areas, e.g. tracking
- iLCSoft / LCIO also proved that common data models help a lot with common software development
- Want to be flexible re. device transfers and offer different persistency options
    - e.g. ALICE Run3 EDM optimised for message passing and the code generation approaches in FCC-hh PODIO EDM generator

Machine Learning

- Machine learning, or artificial intelligence, used for many years in HEP
  - Algorithms learn by example (training) how to perform tasks instead of being programmed
- Significant advances in the last years in ‘deep learning’
  - Deep means many neural network layers
  - Fast differentiability and use of GPUs
- Rapid development driven by industry
  - Vibrant ecosystem of tools and techniques
  - *Highly optimised for modern, specialised hardware*

ML minimisation problem - do this minimisation with 10^6 variables...





An example of a modern ML architecture

Machine Learning in HEP

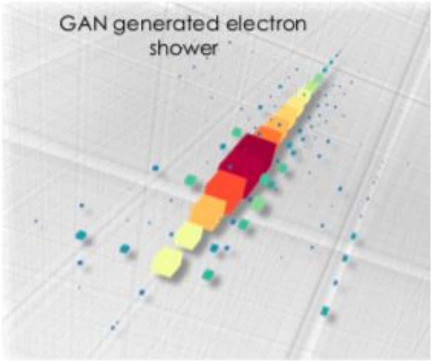
- Better discrimination
  - Important input for analysis (see improvements with Higgs)
  - Also used at HLT as inference can be fast (N.B. training can be slow!)
  - HEP analogies to image recognition or text processing
- Replace expensive calculations with trained output
  - E.g. calorimeter simulations and other complex physical processes
- There are significant opportunities here
  - Need to combine physics and data science knowledge
  - Field evolves rapidly and we need to deepen our expertise
- Integration into our workflows is not at all settled
  - Resource provision, efficient use, heterogeneity and programming models pose problems
  - Training deep models may require *significant* resources

Table 1 | Effect of machine learning on the discovery and study of the Higgs boson

Analysis	Years of data collection	Sensitivity without machine learning	Sensitivity with machine learning	Ratio of P values	Additional data required
CMS[24] $H \to \gamma\gamma$	2011–2012	2.2σ, $P = 0.014$	2.7σ, $P = 0.0035$	4.0	51%
ATLAS[43] $H \to \tau^+\tau^-$	2011–2012	2.5σ, $P = 0.0062$	3.4σ, $P = 0.00034$	18	85%
ATLAS[99] $VH \to bb$	2011–2012	1.9σ, $P = 0.029$	2.5σ, $P = 0.0062$	4.7	73%
ATLAS[41] $VH \to bb$	2015–2016	2.8σ, $P = 0.0026$	3.0σ, $P = 0.00135$	1.9	15%
CMS[100] $VH \to bb$	2011–2012	1.4σ, $P = 0.081$	2.1σ, $P = 0.018$	4.5	125%

Machine learning at the energy and intensity frontiers of particle physics,

https://doi.org/10.1038/s41586-018-0361-2

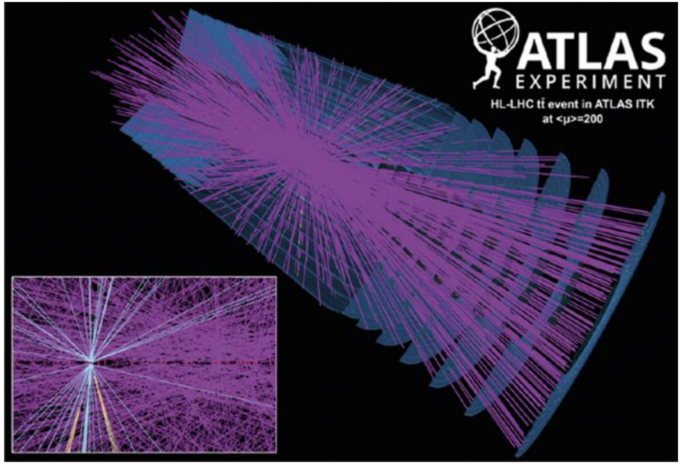


Use of Generative Adversarial Networks to simulate calorimeter showers, trained on G4 events (S. Vallacorsa)

HEP Software and Computing and the HSF Initiative

HEP Software Foundation

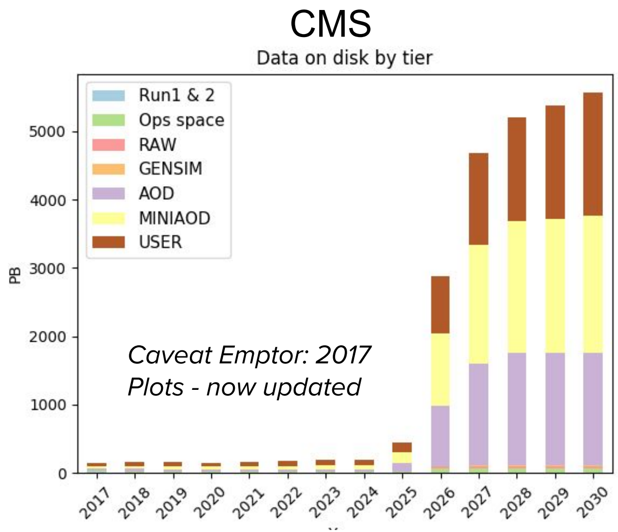
Software at the HL-LHC

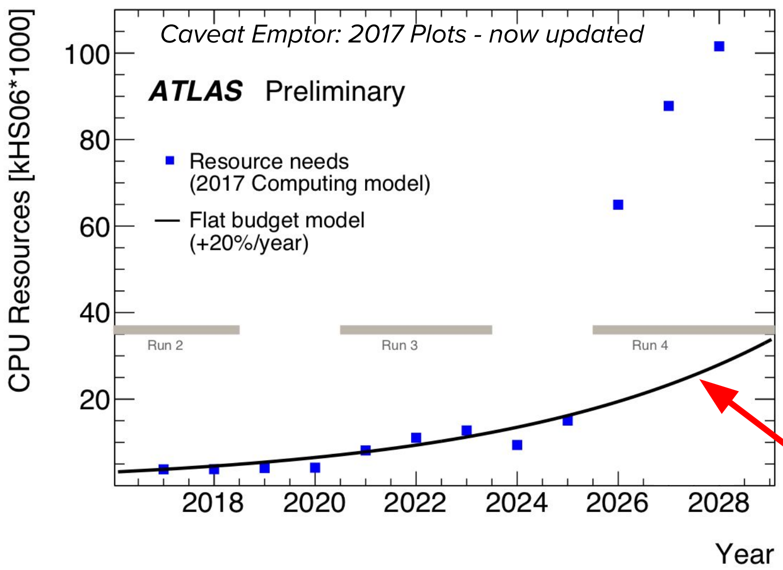


- Pile-up of ~200 ⇒ particularly a challenge for charged particle reconstruction
  - Inner trackers and CMS High Granularity Calorimeter
- HEP software typically executes one instruction at a time (per thread)
  - Since ~2013 CPU (core) performance increase is due to more internal parallelism
  - x10 with the same HW only achievable if using the full potential of processors
    - Major SW re-engineering required (but rewriting everything is not an option)
  - Co-processors like GPUs *require* that this problem is solved
- Increased amount of data requires to revise/evolve our computing and data management approaches
  - We must be able to feed our applications with data efficiently
- *HL-LHC salvation will come from software improvements, not from hardware*

Challenges for the Next Decade

- HL-LHC brings a huge challenge to software and computing
  - Both rate and complexity rise





- Not just a simple extrapolation of Run 2 software and computing
  - Resources needed would hugely exceed those from technology evolution alone

[This is probably too optimistic]

HEP Software Foundation (HSF)

- 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 parallelisation
  - New approaches needed to overcome limitations in today's code
- Cannot afford any more duplicated efforts
  - Each experiment has its own solution for almost everything (framework, reconstruction algorithms, …)
- HSF started with a number of workshops and working groups on common topics (packaging, licensing)
- The goal of the HSF is to facilitate coordination and common efforts in software and computing across HEP in general
  - Our philosophy is bottom up, a.k.a. *do-ocracy*

Community White Paper Inception

- We wanted to describe a **global vision for software and computing** for the HL-LHC era and HEP in the 2020s

- Formal charge from the WLCG in July 2016
  - Anticipate a "software upgrade" in preparation for HL-LHC
  - Identify and prioritize the software research and development investments
    1. to achieve improvements in software efficiency, scalability and performance and to make use of the advances in CPU, storage and network technologies
    2. to enable new approaches to computing and software that could radically extend the physics reach of the detectors
    3. to ensure the long term sustainability of the software through the lifetime of the HL-LHC

- Long process of 1 year, with many working groups and 2 major workshops

A Roadmap for HEP Software and Computing R&D for the 2020s

HSF-CWP-2017-01
December 15, 2017

- 70 page document
- 13 sections summarising R&D in a variety of technical areas for HEP Software and Computing
  - Almost all major domains of HEP Software and Computing are covered
- 1 section on Training and Careers
- 310 authors from 124 institutions
- https://doi.org/10.1007/s41781-018-0018-8; arXiv:1712.06982

Contents

```
int main {
  cout << “write software” << endl;
  return 0;
}
```

HSF Working Groups

- The Roadmap established what challenges the community faced
  - But it did not spell out *how* to face them in detail
- HSF had adopted a model of working groups from its earliest days
  - These were open groups of people in the community, motivated enough to organise around a common topic, usually at their own initiative
- This model seemed a good one for moving forwards on the key topics
  - We were a little more formal this time around
    - Call for nominations from the whole community, then search committee
    - Significant engagement from LHC experiments and beyond, e.g. Belle II
- The HSF's role is one of an information conduit and meeting point
  - Report on interesting and common work being done
  - Forum for technical comments and discussion
  - Encourage cooperation across experiments and regions

Some important practical matters! Copyright and Licensing

Software Licence Agreements
HSF Policy Guidelines

[HSF-TN-2016-01]

M. Jouvin[1] J. Harvey[2] A. McNab[3] E. Sexton-Kennedy[4] T. Wenaus[5]

[1] *Laboratoire de l'Accélérateur Linéaire (CNRS)* [2] *CERN* [3] *University of Manchester* [4] *Fermi National Accelerator Laboratory* [5] *Brookhaven National Laboratory*

- Long neglected inside collaborations
  - Code was arbitrarily licensed or unlicensed, copyright assigned to random authors and institutes
  - Yet this is essential to be able to
    - Open source our software properly
    - Combine with other open source projects and collaborate
- Copyright
  - Advice to keep this as low a number as practicable as copyright holders decide the licence
  - LHC experiments: © CERN for the benefit of collaboration X
- License
  - Favour liberal licenses for industry collaboration: LGPL, Apache, MIT
  - Definitely avoid GPL for libraries you want other people to use

Software Nuts and Bolts

HSF Packaging Working Group Report

[HSF-TN-2016-03]

L. Sexton-Kennedy[1], B. Hegner[2], B. Viren[3]

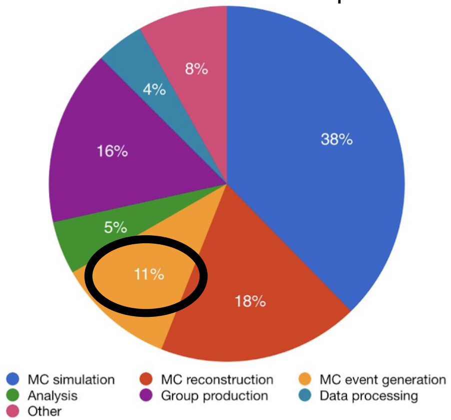
[1]FNAL,[2]CERN,[3]BNL

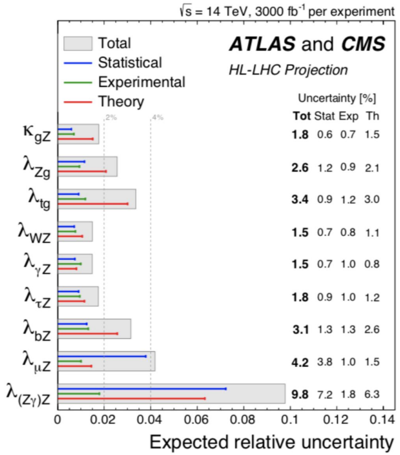
- Software Tools WG
  - Active group promoting best practice for correctness and performance
  - There has been a revolution in adopting best open source practice in recent years
    - git, GitHub, GitLab, CMake, merge requests, code review, ...
  - HSF has an active group promoting best practice for correctness and performance
    - Profiling, static analysis
- Packaging WG
  - We don’t build our experiment software in isolation
  - Need a software stack, incorporating many components from the open source world and HEP community
    - This touches deeply on license and license combinations
  - Preference for tools that are not home grown and have a wider support base
  - Spack (LBNL) and Conda actively being prototyped

Event Generators

- Event generators are the start of the simulation chain
  - At the LHC Run1 only leading order generators were used
  - Negligible CPU consumption compared with detector simulation - no pressure to optimise
- However, with LHC upgrades coming higher order generators become much more important
  - These are inherently much more costly to run
  - Problems of negative weights can increase hugely the samples needed for weighted event samples
- In addition, the theory community, who develop these codes usually work in small teams
  - Recognition for technical improvements is limited/missing

ATLAS 2018 CPU Report

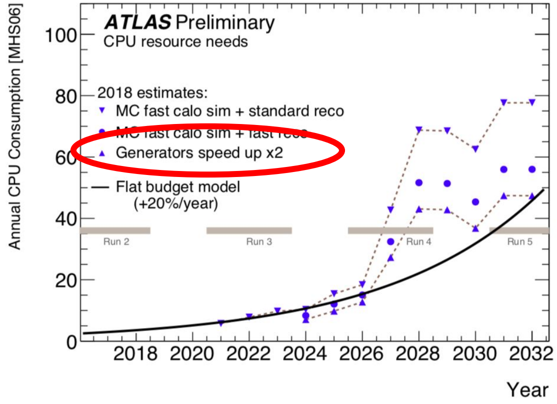


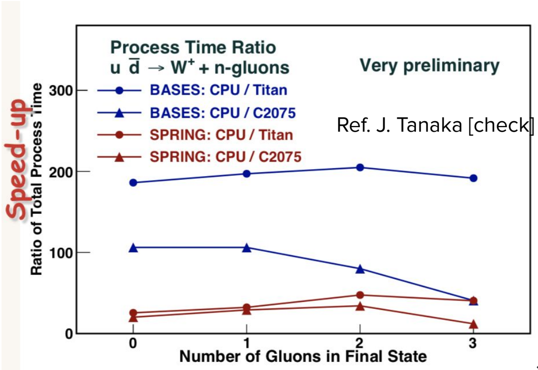


Many electroweak measurement errors dominated by theory (red). B. Hinemann

Event Generators - Technical Improvements

- HSF/LPCC workshop in November brought theory and experiment together to look at computing challenges of event generation
    - This was the first workshop of its kind
- Working group tackling technical challenges
    - Setting a baseline for further comparisons
    - Understanding how to run generators for best efficiency
    - Support for technical improvements (e.g. thread safety)
    - Porting to other architectures
        - Could be very suitable code to do this with (smaller, self contained code bases, numerically intensive)
        - e.g. building on the work done so far in MadGraph with GPUs

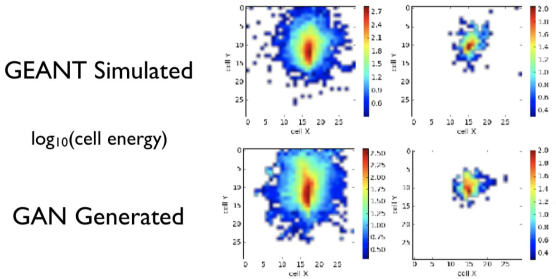




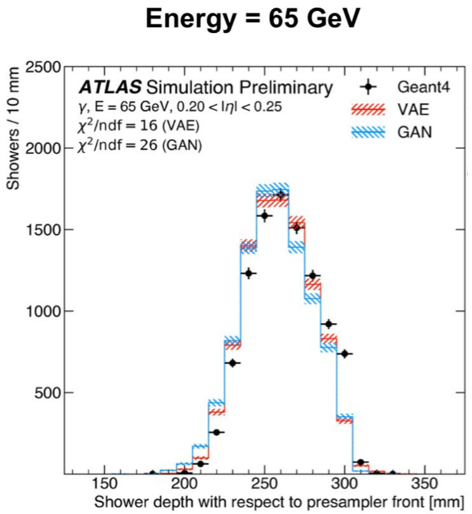
Ref. J. Tanaka [check]

Detector Simulation

- A major consumer of LHC grid resources today
    - Experiments with higher data rates will need to more simulation
- Faster simulation, with no or minimal loss of accuracy, is the goal
    - Range of techniques have been used for a long time (frozen showers, paramtric response)
    - Key point is deciding when it’s good enough for physics
- Machine learning lends itself to problems like this
    - Calorimeter simulations usually targeted
    - Variational Auto Encoders (VAEs) attempt to compress the data down to a ‘latent space’ - can be randomly sampled to generate new events
    - Generative Adverserial Networks (GANs) train two networks, one to generate events, the other to try to classify as real/fake
    - R&D on lifecycle integration into Geant4 is starting...

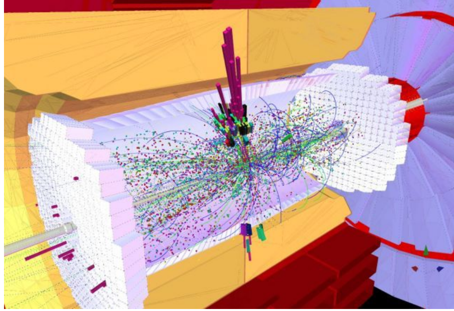


LHCb ECal simulated with G4, generated with GAN [F. Ratnikov]



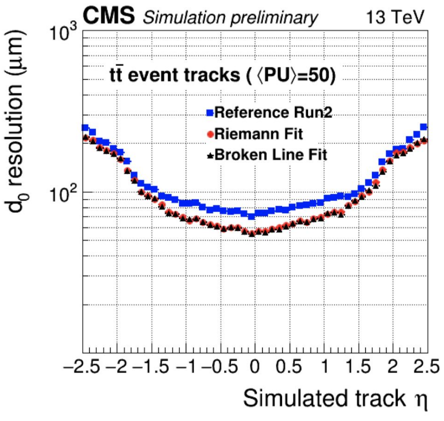
ATLAS VAE and GAN cf. Geant4 simulation [ATL-SOFT-PUB-2018-001.]

Detector Simulation



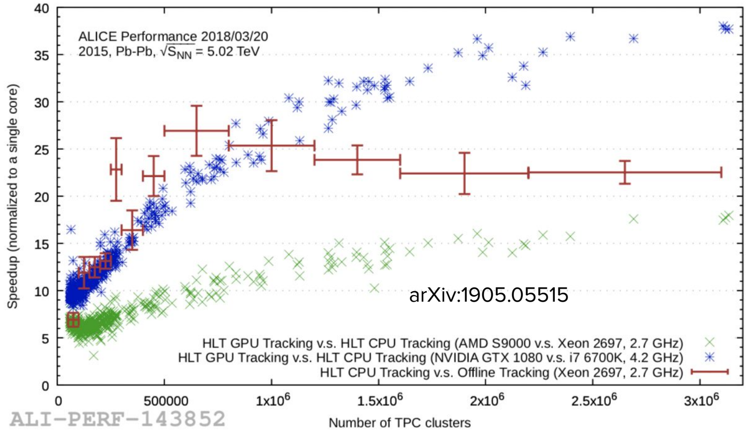
- Technical improvement programme helps (and helps *everyone*)
- GeantV R&D modernises code and introduces vectorisation
  - Speed-ups observed
  - Vectorisation introduces small gains
  - Code modernisation seems to help a lot
- Geant4 now have a new R&D working group that will take studies forward
- Some studies of running Geant4 on GPUs have begun
  - US Exascale Computing Project is funding this
  - Motivated by the next generation of US supercomputers that target exaflop
    - 90-95% of FLOP capacity in GPUs
  - However, migration of physics code is an incredibly tricky business
    - This would be a long haul, but a huge achievement for all of HEP

Reconstruction and Software Triggers



(a) d_0 resolution vs η

- Hardware triggers no longer sufficient for modern experiments
  - More and more initial reconstruction needs to happen in software
- Close to the machine, need to deal with tremendous rates and get sufficient discrimination
  - Pressure to break with legacy code is high
  - Lots of experimentation with rewriting code for GPUs
    E.g. LHCb's Allen project (HLT1 on GPU)
  - ALICE have ported a lot of reconstruction to GPUs and also improved the algorithms a lot
  - CMS Patatrack project has improved physics performance as well
    - Revisiting old code helps!
- Lessons learned keep data model simple, bulk data, be asynchronous, minimise data transfers

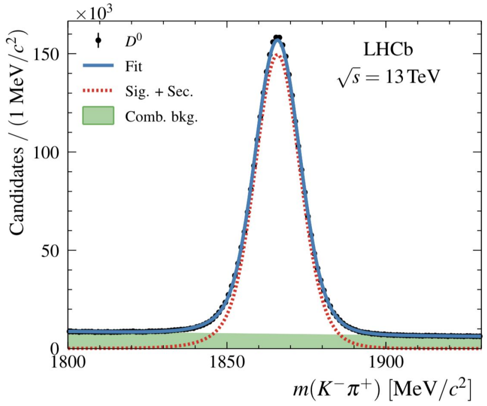


Reconstruction and Software Triggers

- Real Time Analysis (HEP Version)
  - Design a system that can produce analysis useful outputs as part of the trigger decision
    - If this captures the most useful information from the event, can dispense with raw information
  - *This is a way to fit more physics into the budget*
- LHCb Turbo Stream has been introduced in Run2 and will be dominant in Run3
- Whole ALICE data reduction scheme is based around keeping 'useful' parts of events (no more binary trigger)
  - O2 → Online/Offline Data Reduction Farm
- ATLAS and CMS have schemes under development for special handling of samples for which full raw data is unaffordable (aka. data scouting)

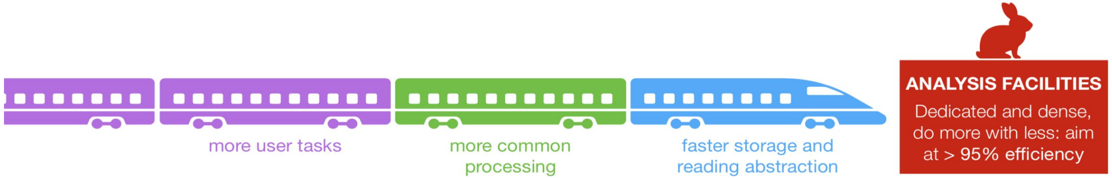
Persistence method	Average event size (kB)
Turbo	7
Selective persistence	16
Complete persistence	48
Raw event	69

LHCb Run2 Turbo took 25% of events for only 10% of bandwidth

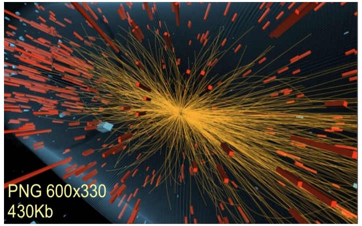


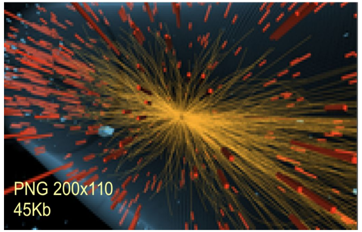
LHCb charm physics analysis using Turbo Stream (arXiv:1510.01707)

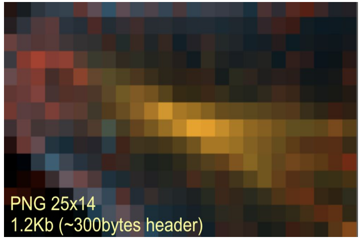
Analysis



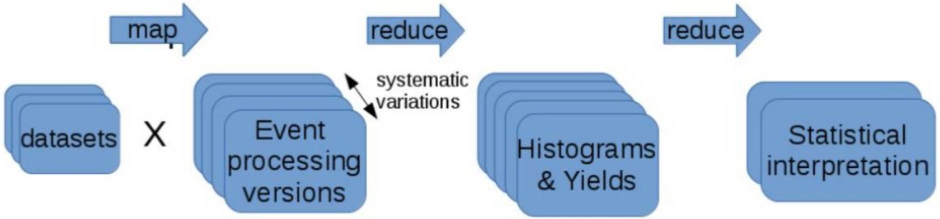
- Scaling for analysis level data also a huge challenge for all LHC experiments
- Efficient use of analysis data can come with combining many analyses as carriages in a train like model (pioneered by PHENIX and then ALICE)
  - Also goes well with techniques like tape carousels (ATLAS scheme for rotating primary AOD data from tape systems into a disk buffer)
  - Interest in *analysis clusters*, specialised for analysis operations over the generic grid resources (WLCG/HSF pre-CHEP workshop 2-3 November)
- Reducing volume of data needed helps hugely
  - CMS ~1kB nanoAOD makes a vast difference to analysis efficiency and “papers per petabyte”
  - Smaller EDM is easier to make efficient
  - Requires analyst agreement on corrections, scale factors, etc.
    - However the alternative is perhaps that your analysis never gets done







Analysis



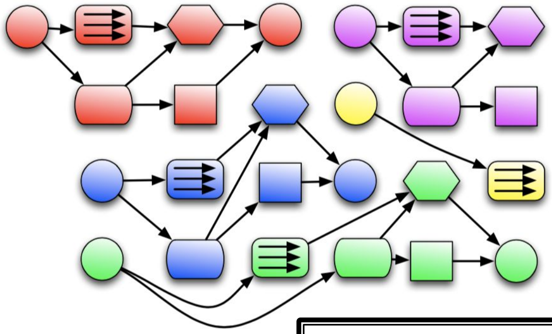
- Improve analysis ergonomics - how the user interacts with the system to express their analysis
  - Streamline common tasks
    - Handle all input datasets; Corrections and systematics
    - Compute per event and accumulate; Statistical interpretations
  - Declarative models, building on ROOT's RDataFrame
    - Say *what*, not *how* and let the backend optimise
    - E.g. split and merge, GPU execution
- Notebook like interfaces gain ground, as do containers - lots of high level Python
- Interest in data science tools and machine learning is significant for this community - inspiring new approaches (e.g. uproot, awkward array, scikit-hep, Coffea)
  - This is an ecosystem into which HEP can contribute

Many analysis frameworks, multiple per experiment, not well generalised

```
# * Jet select/cleaning against loose leptons , jet pt > 25 , jet id
flow.DefaultConfig(jetPtCut=25,jetIdCut=0,jetPUIdCut=0)
flow.SubCollection("CleanJet","Jet",'''
 Jet_pt > jetPtCut &&
 Jet_jetId > jetIdCut &&
 Jet_puId > jetPUIdCut &&
 (Jet_LeptonIdx==-1 || Jet_LeptonDr > 0.3)
''')
```

A. Rizzi, NAIL prototype

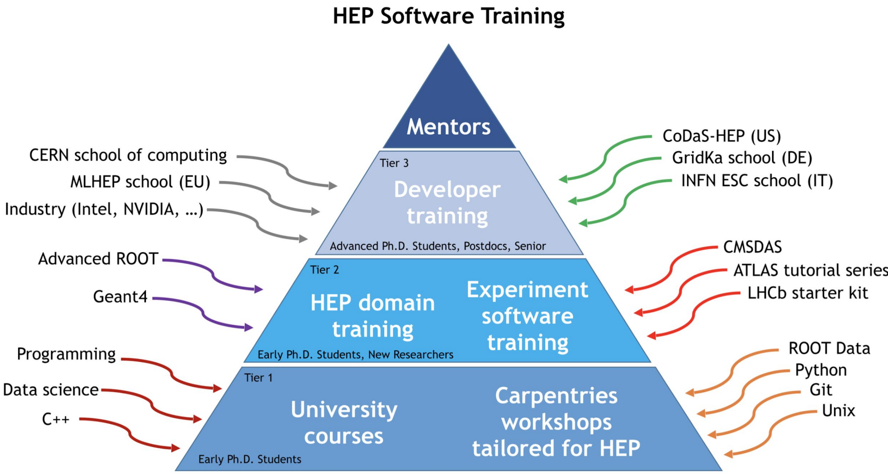
Frameworks and Integration



Cartoon of a single job, processing multiple events (colours) through different modules (shapes)

- Increasingly heterogeneous world requires advanced software support infrastructure
  - Software frameworks support use of different devices as well as insulate developers from many of the details of concurrency and threading models
    - Adapt to the new heterogeneous landscape
    - Latency hiding is critical to maintaining throughout
  - Framework development has traditionally been quite fragmented, but new experiments should offer a chance to increase convergence
    - Better to start off together than try to re-converge later (iLCSoft, LArSoft examples of success, albeit without concurrency; Gaudi for LHCb, ATLAS)
    - ALFA for ALICE and FAIR experiments
- New HSF working group being established now (draft mandate)

Training and Careers



- Many new skills are needed for today's software developers and users
- Base has relatively uniform demands
  - Any common components help us
- LHCb StarterKit initiative taken up by several experiments, sharing training material
  - Links to 'Carpentries' being remade (US training projects) - up the level!
- New areas of challenge
  - Concurrency, accelerators, data science
  - Need to foster new C++ expertise (unlikely to be replaced soon as our core language, but needs to be modernised)
- Careers area for HEP software experts is an area of great concern
  - Need a functioning career path that retains skills and rewards passing them on
  - Recognition that software is a key part of HEP now

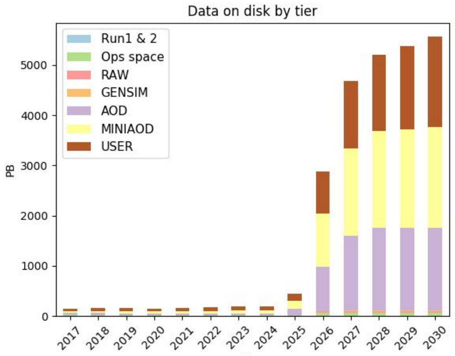
European Strategy Update

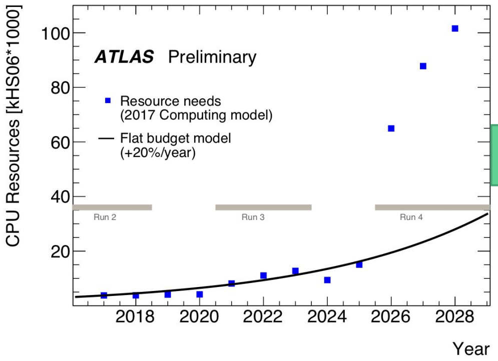
Instrumentation and Computing

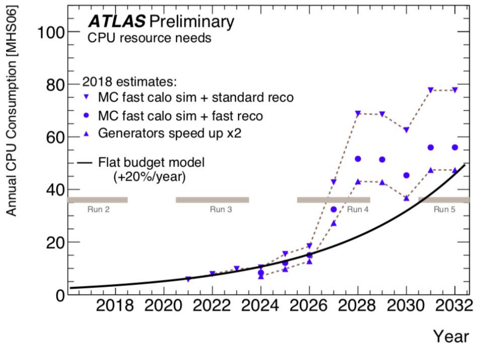
Xinchou Lou, Brigitte Vachon
Scientific Secretaries: Emilia Leogrande, Rogers Jones

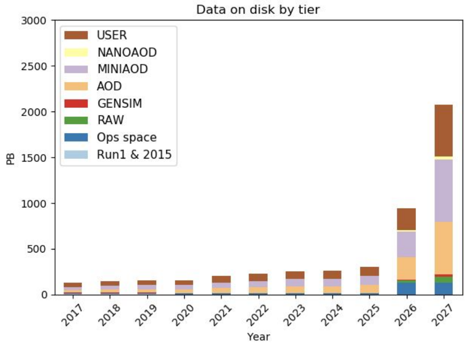
Meeting the HL-LHC Challenge!

- Already since the Roadmap was written experiments have made great progress in meeting the HL-LHC challenge
  - Bad software, is extremely expensive
  - Good and clever software allows much more physics to fit in the budget









Conclusions

- We have a wide ranging and ambitious physics programme in HEP and in associated disciplines
    - Our experiments are highly data intensive and require high quality software and computing
- The landscape for software is becoming ever more challenging
    - Working together on common problems is not only the best use of our resources, our funding agencies will mandate it
- HSF is now established to help HEP achieve that goal and marshalls effort around the community
    - Roadmap delivered and active working groups in key areas

HL-LHC is a challenge and also a great opportunity to improve HEP software

HSF Getting Involved...

- Join the HSF Forum, hsf-forum@gmail.com
  - Few messages a week with updates, jobs, items of interest
  - Owned by the community - please just post items of relevance
- Join a working group, https://hepsoftwarefoundation.org/what_are_WGs.html
  - Follow the group's meetings and discussions
  - Suggest a meeting topic
- Annual meetings and Workshops
  - Establishing a tradition of a joint meeting with WLCG each Year (next short meeting pre-CHEP, November)
- Propose a new activity area
  - The HSF is there to help gather interest

- Data Analysis
- Detector Simulation
- Frameworks
- Physics Generators
- Packaging
- PyHEP - Python in HEP
- Quantum Computing
- Reconstruction and Software Triggers
- Software/Developer Tools
- Training
- Visualization