HEP Software Foundation

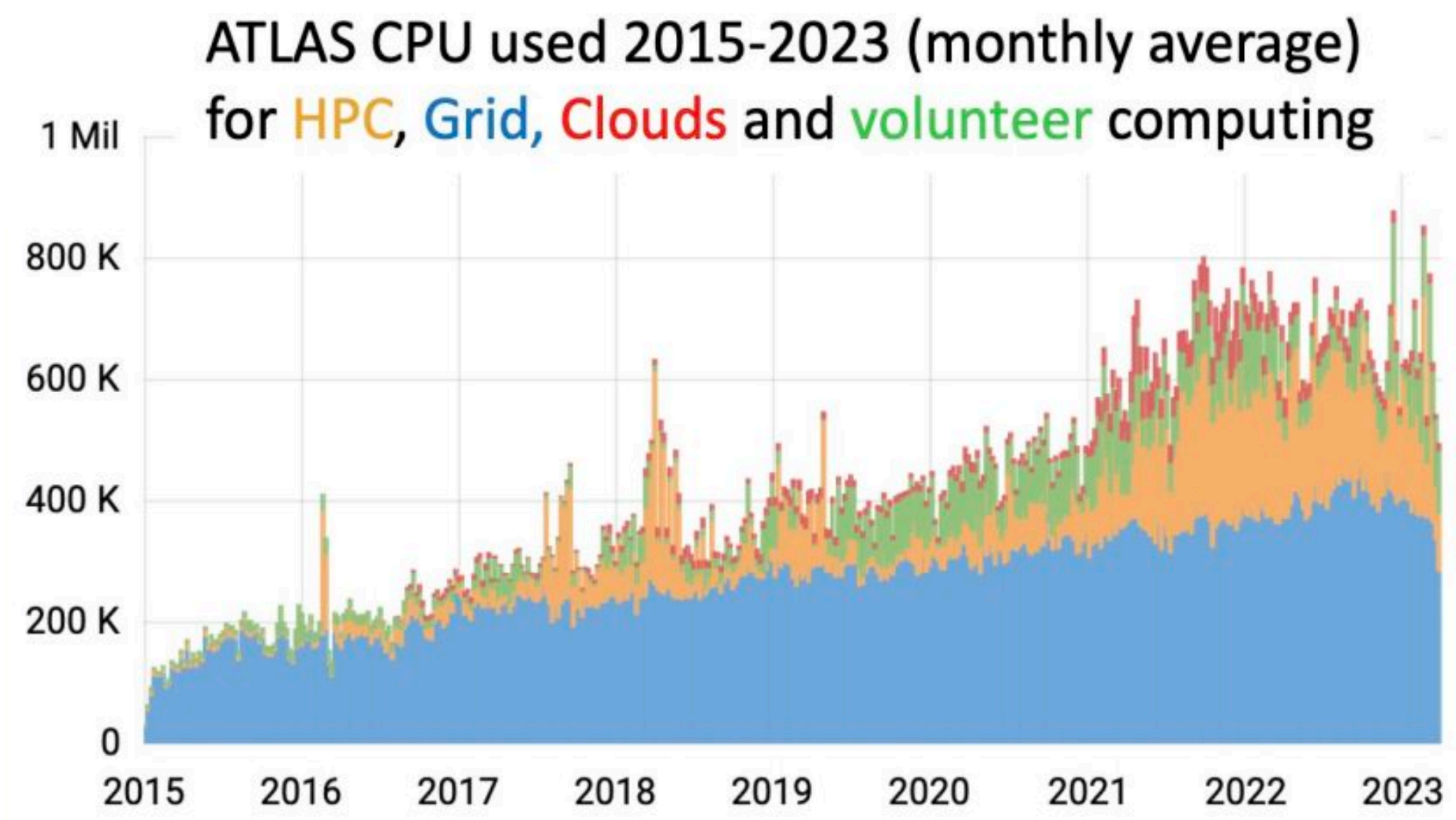
Software and the Interface to Computing

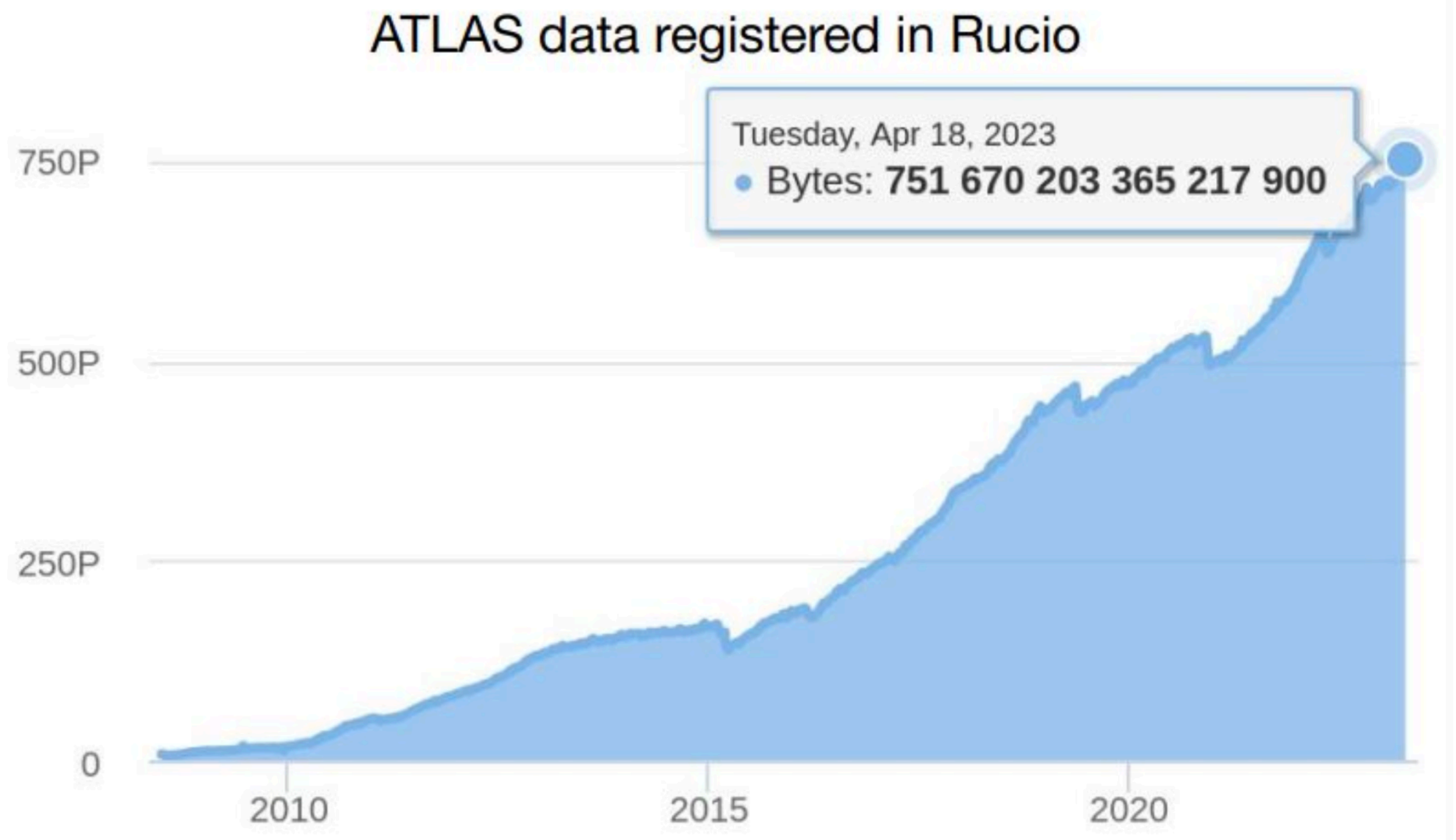
Graeme Stewart, 2023-10-27

ET-PP/ET-EIB Workshop

The LHC Example: ATLAS Software and Computing in a nutshell…

- Currently in LHC Run-3 ATLAS has
  - Steady 600k CPUs, peaks of 1M+
  - 750PB of data, 1B+ files
  - Experiment production software base around 6M+ lines (mainly C++ and Python)
  - Resting on many millions of lines of other HEP codes (Geant4, ROOT, Pythia, Awkward, etc.)
- In total HEP probably has 50M+ lines of code
  - Would cost ~€500M+ to develop commercially
- For High Luminosity LHC (2029)
  - Trigger rate rises from 1.5kHz to ~10kHz - event rate ⬆
  - Pile-up increases from ~50 towards ~200 - event complexity ⬆


ATLAS CPU used 2015-2023 (monthly average) for HPC, Grid, Clouds and volunteer computing


ATLAS data registered in Rucio

athena
ATLAS Experiment main repository for Athena code

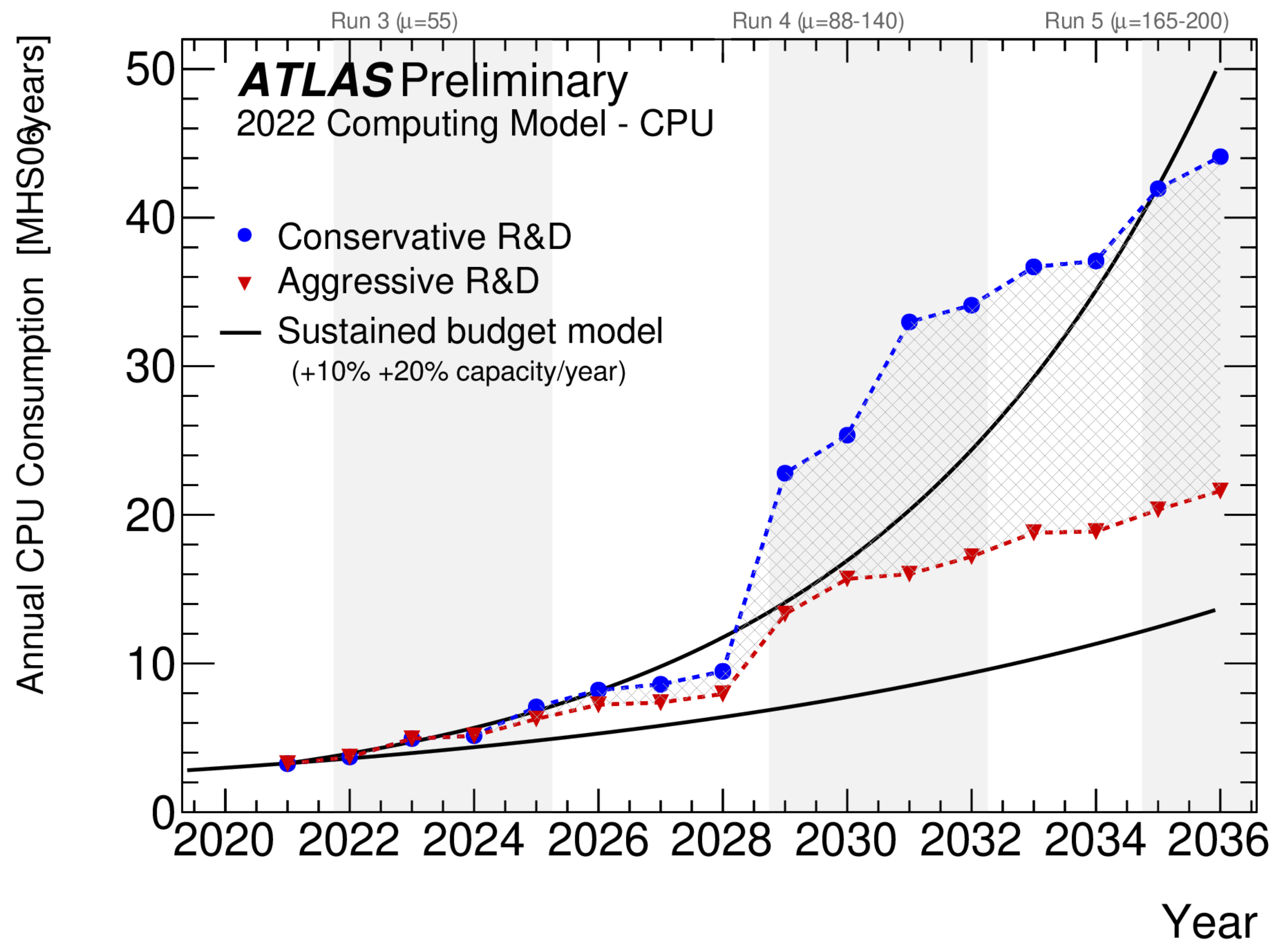
ROOT
Data Analysis Framework

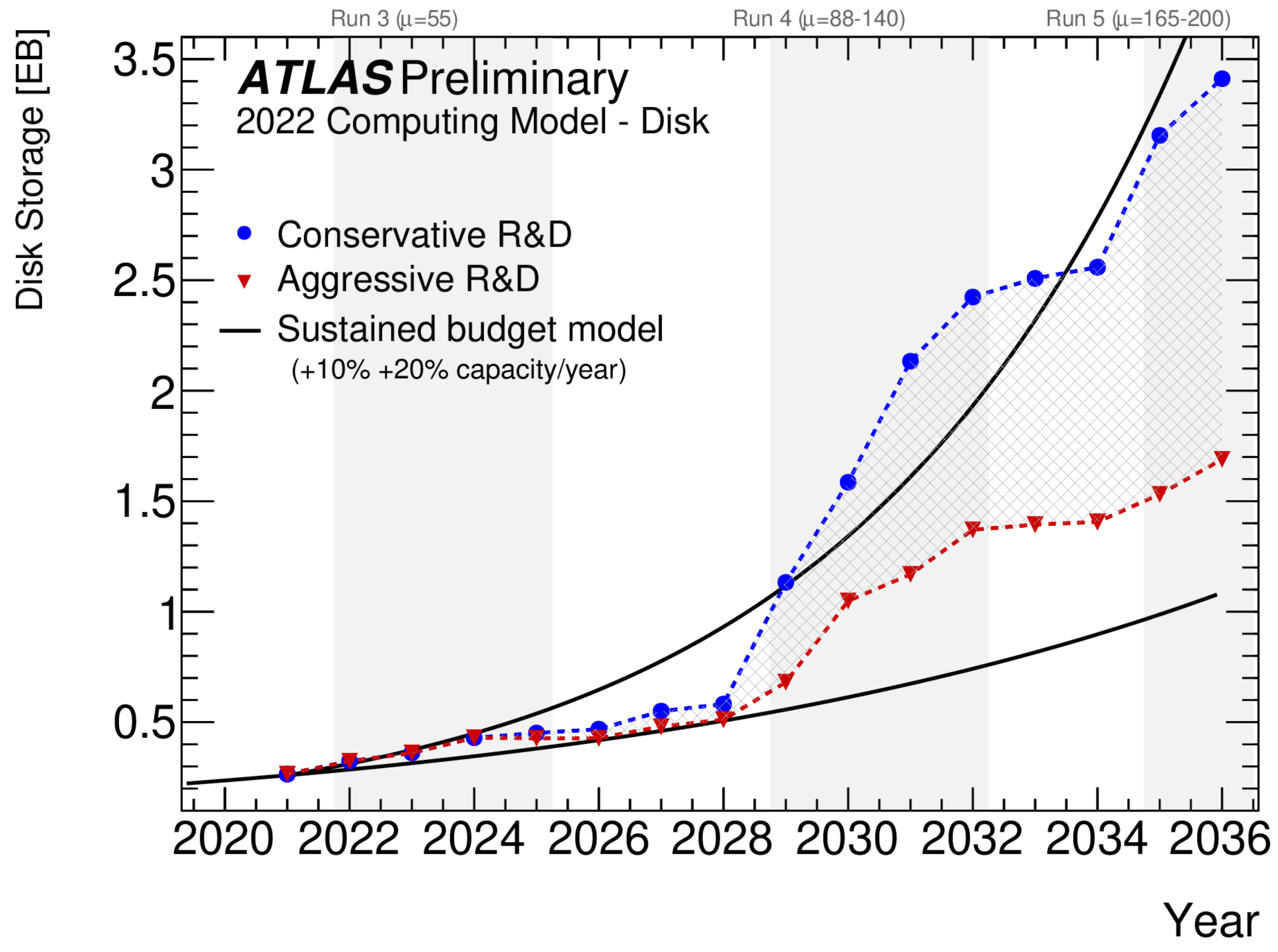
GEANT4
A SIMULATION TOOLKIT

Scikit HEP

The LHC Example: ATLAS Software and Computing in a nutshell…

- Currently in LHC Run-3 ATLAS has
  - Steady 600k CPUs, peaks of 1M+
  - 750PB of data, 1B+ files
  - Experiment production software base around 6M+ lines (mainly C++ and Python)
  - Resting on many millions of lines of other HEP codes (Geant4, ROOT, Pythia, Awkward, etc.)
- In total HEP probably has 50M+ lines of code
  - Would cost ~€500M+ to develop commercially
- For High Luminosity LHC (2029)
  - Trigger rate rises from 1.5kHz to ~10kHz - event rate ⬆
  - Pile-up increases from ~50 towards ~200 - event complexity ⬆



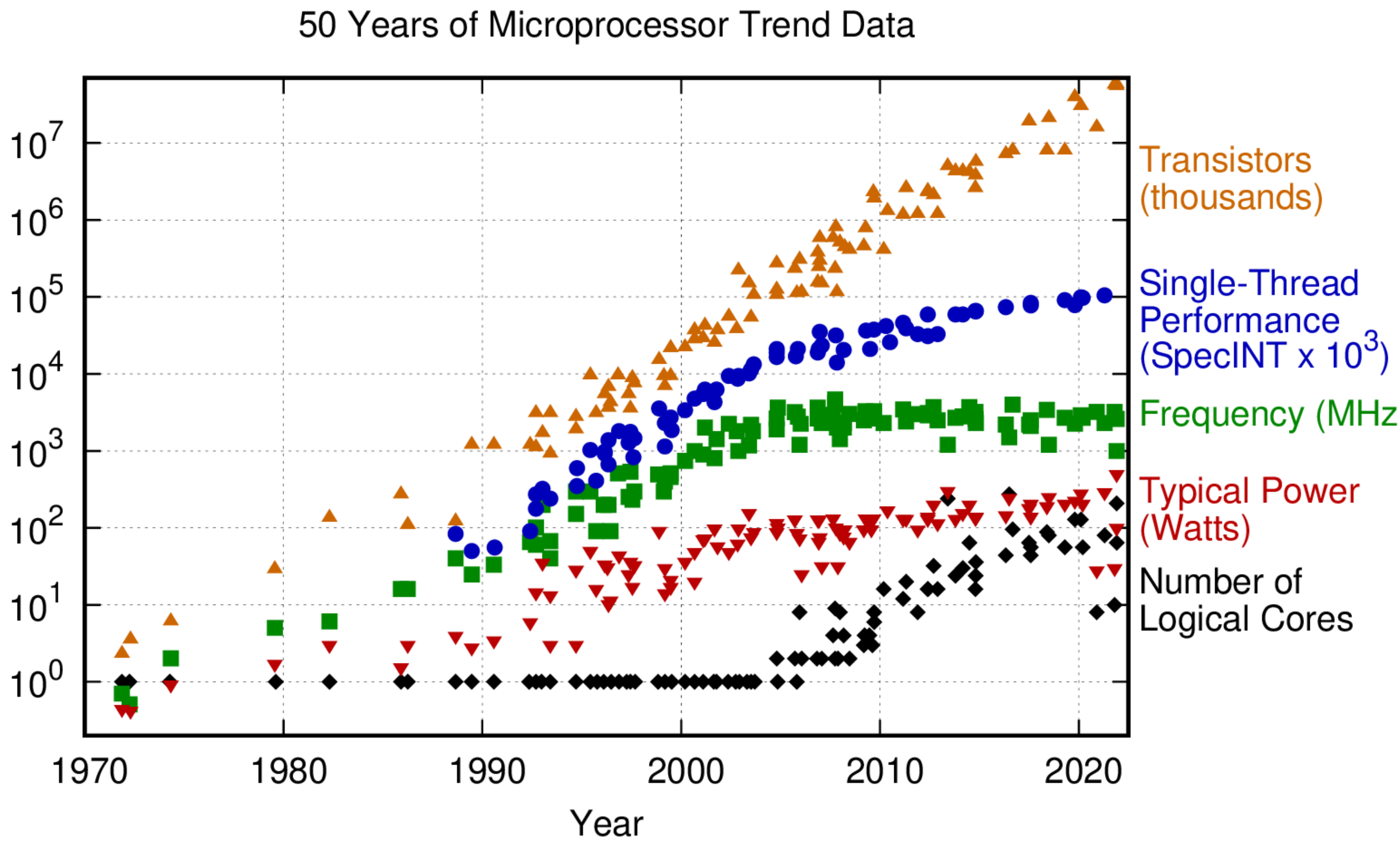


ATLAS Experi

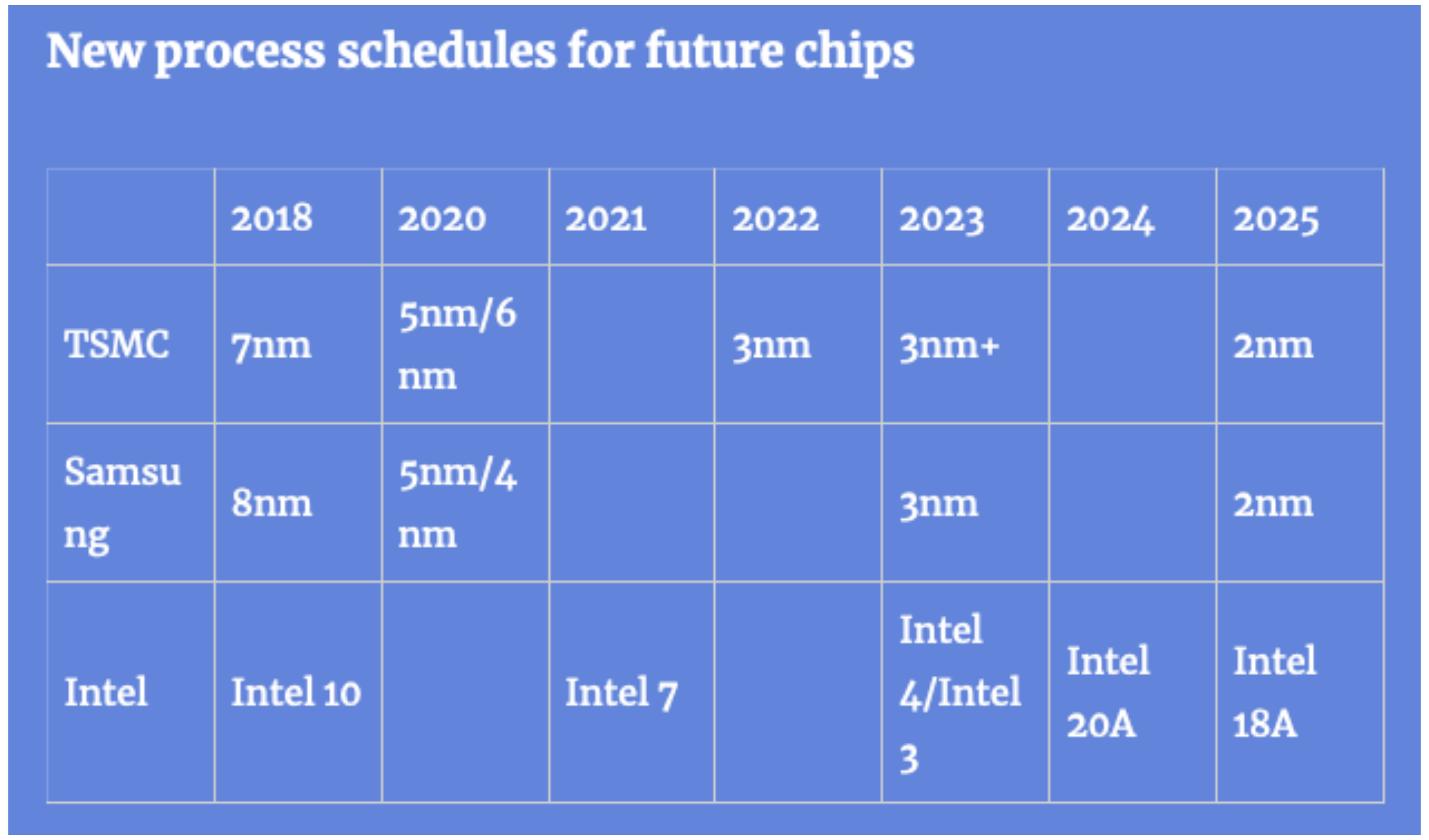
Processor Hardware and Technology

CMOS Transistors

- Moore’s Law continues to deliver increases in transistor density - at least for now!
  - Increasingly challenging technical issues, but there is a roadmap to 2nm* by ~2025
  - Transistors’ smallest scales now consist of very few atoms (10-100), so we are in the endgame
- Clock speed scaling failed many years ago
  - 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 ~100s of clock cycles


50 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 2010-2021 by K. Rupp

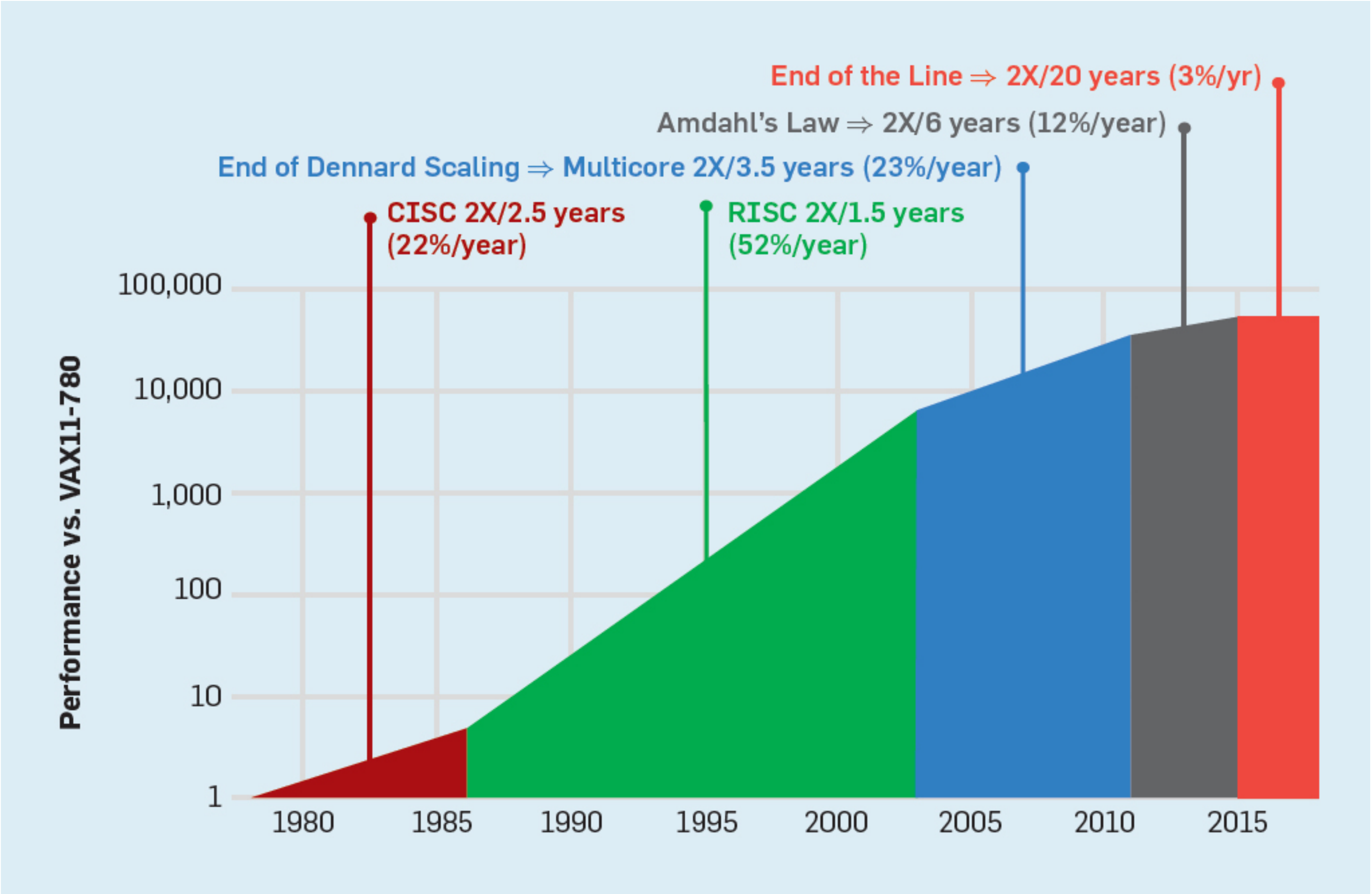

New process schedules for future chips

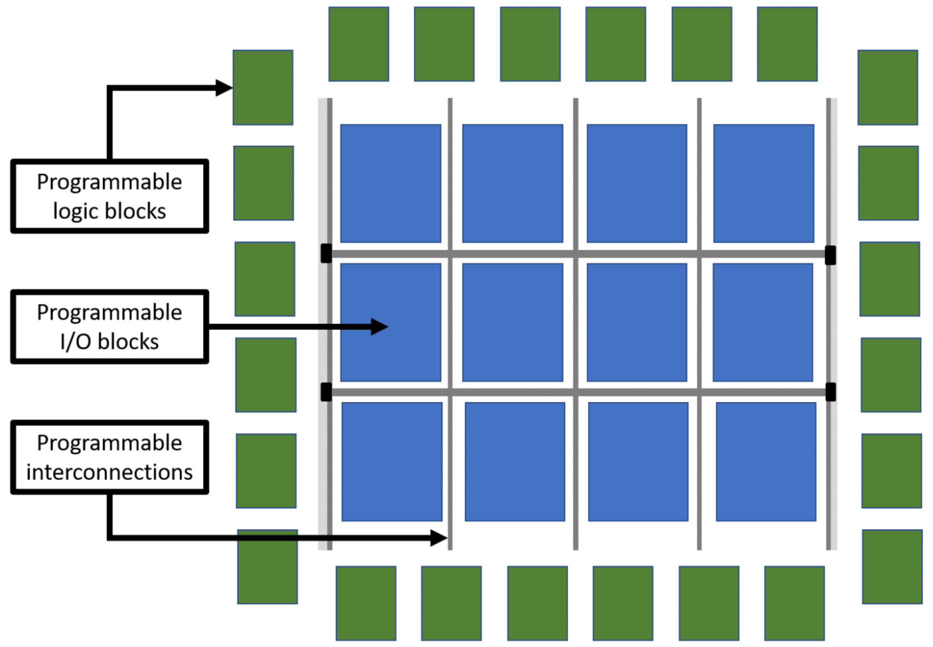
	2018	2020	2021	2022	2023	2024	2025
TSMC	7nm	5nm/6nm		3nm	3nm+		2nm
Samsung	8nm	5nm/4nm			3nm		2nm
Intel	Intel 10		Intel 7		Intel 4/Intel 3	Intel 20A	Intel 18A

*Caveat emptor - the “process size” is not a meaningful physical length

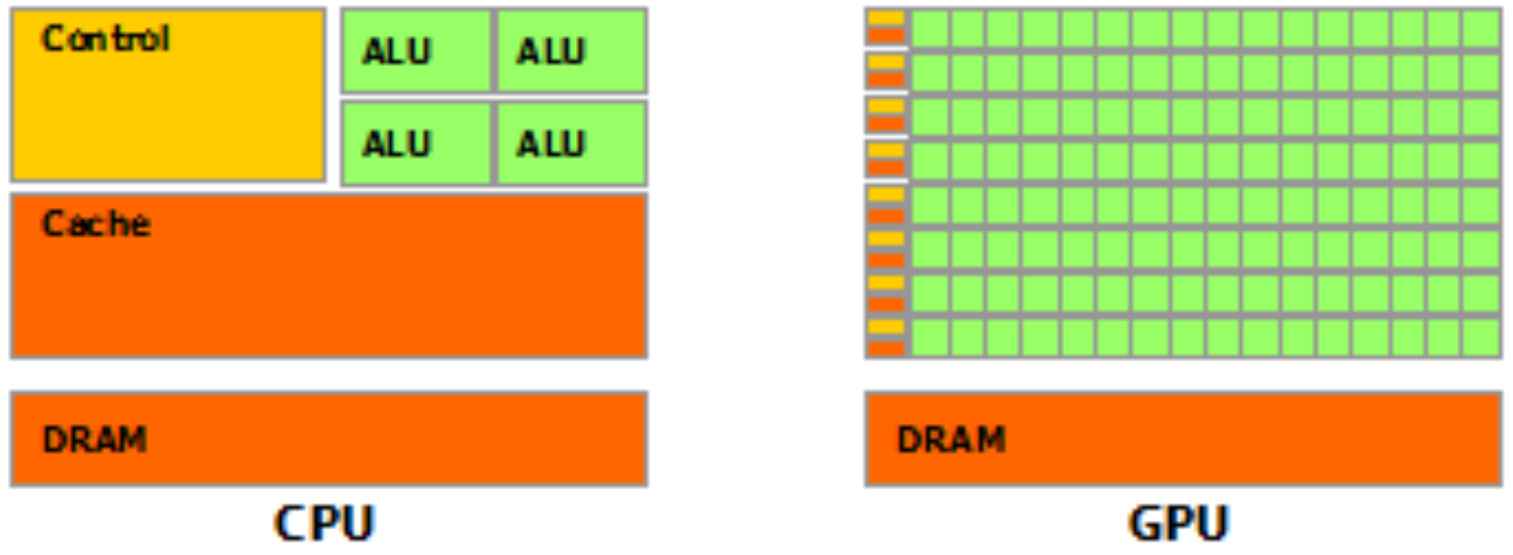
Decreasing Returns and Diversity

- Diversity of new architectures will only grow
  - Chiplets technique enables “Lego” style custom chips
- Best known example is of GPUs
  - Also FPGAs, TPUs
- As well as non-trivial innovations for CPUs
  - Apple M2/3
  - Ampere Altra and Graviton
  - Fujitsu A64FX
  - Google Tensor
- ARM architectures gain about 30% in power efficiency on HEP workloads





FPGAs implement data and logic flow directly on their hardware



GPUs dedicate far more transistor area to arithmetic calculations

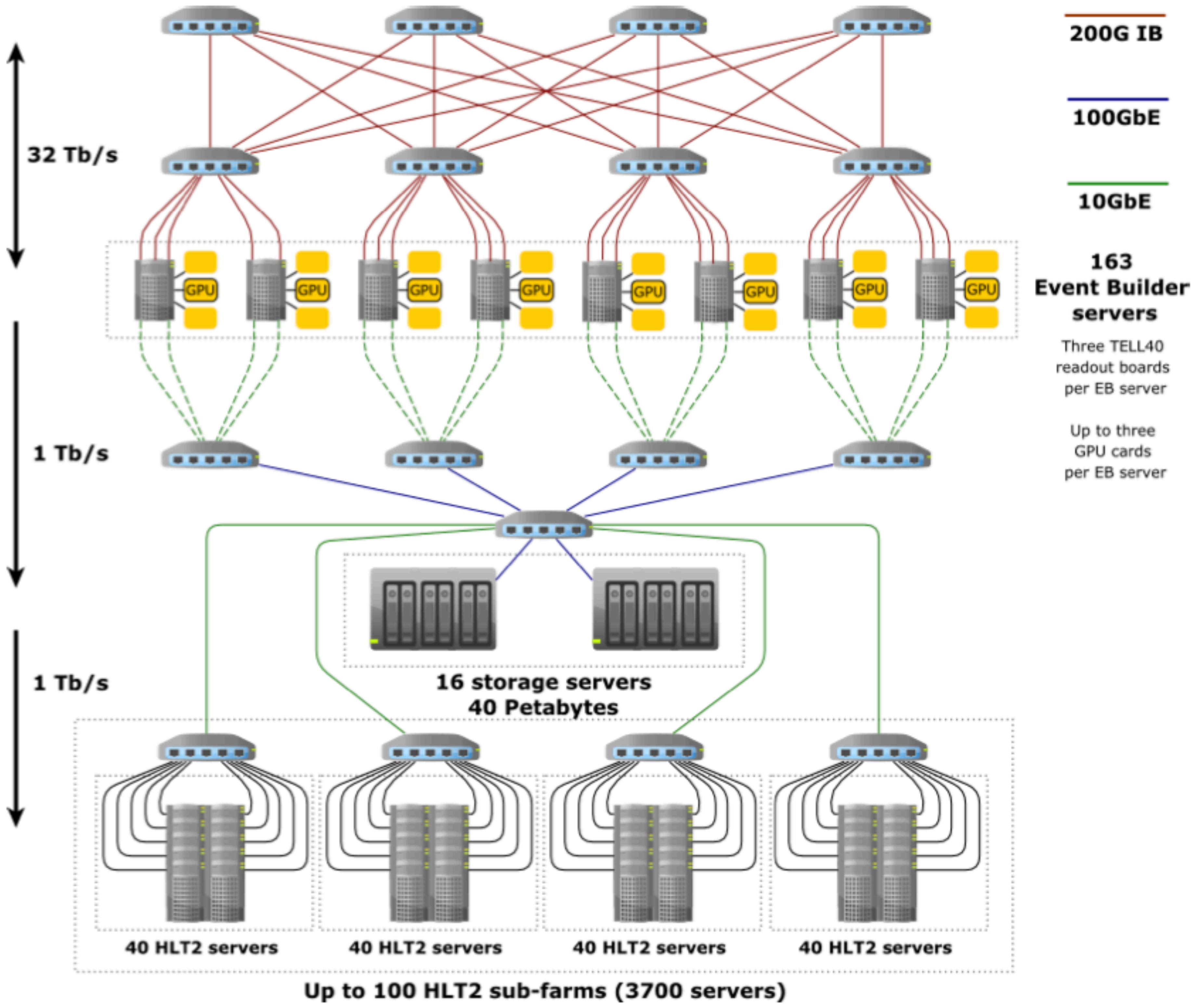
Trends in (HEP) Physics Software

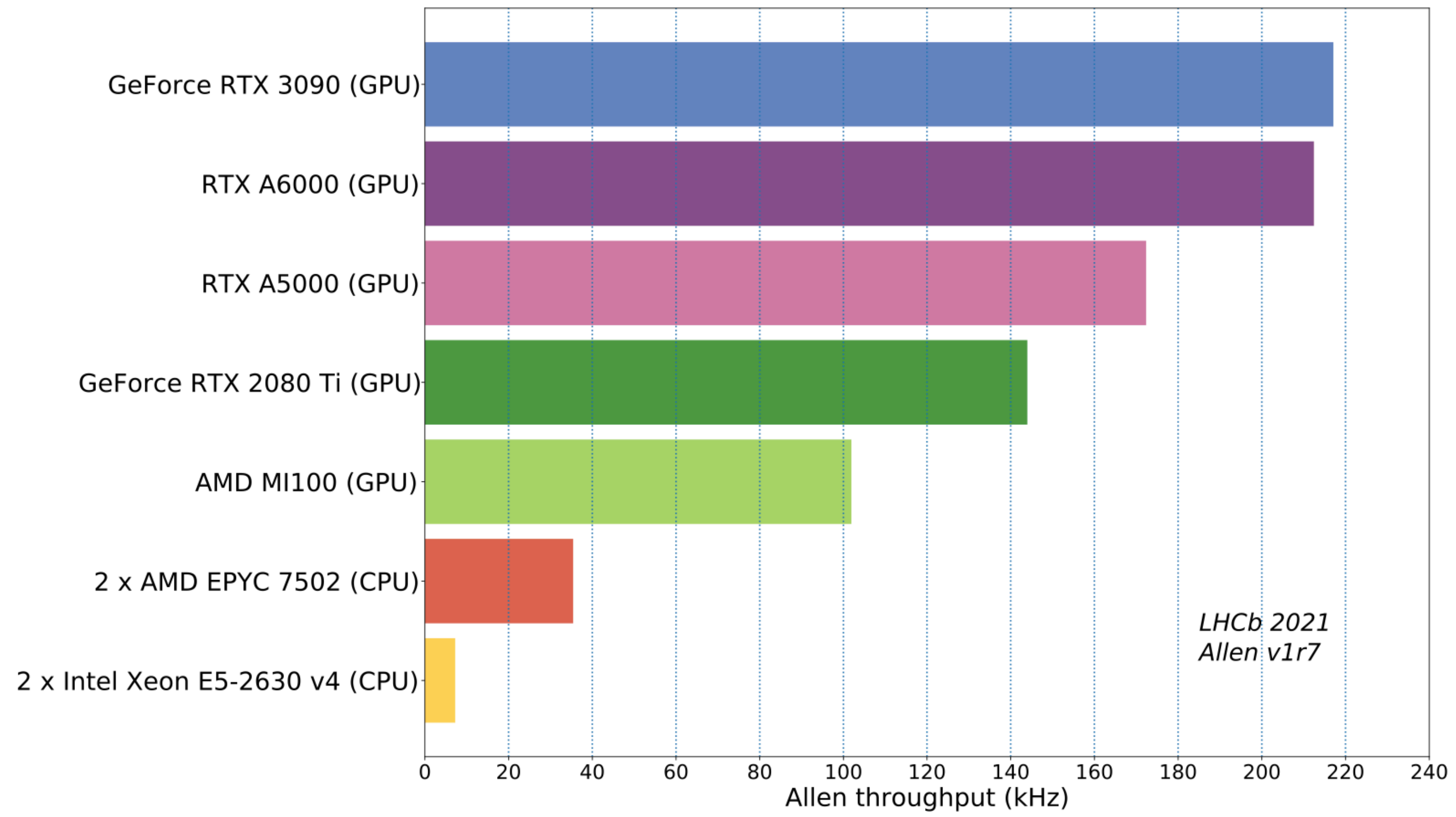
GPU Case Study: LHCb Allen Framework

- LHCb studies flavour physics where signal rates are extremely high
    - Traditional binary triggers are not effective - categorise different signals fast, need *access to as much of the event as possible at 30MHz*
- GPUs are a good fit for LHCb events, if used as *primary processors*, rather than coprocessors
    - Challenge was to convert the *whole HLT1 workload* to a GPU workflow
    - Hosting the GPU cards in the event builder nodes reduces costs significantly

Need to parallelise data processing algorithms and do as much processing as possible on the GPU

Caveat Emptor: Not easy or efficient for all workflows


200G IB
100GbE
10GbE
32 Tb/s
1 Tb/s
1 Tb/s
GPU
163 Event Builder servers
Three TELL40 readout boards per EB server
Up to three GPU cards per EB server
16 storage servers
40 Petabytes
40 HLT2 servers
40 HLT2 servers
40 HLT2 servers
40 HLT2 servers
Up to 100 HLT2 sub-farms (3700 servers)



Ref: https://indico.jlab.org/event/459/contributions/11817/

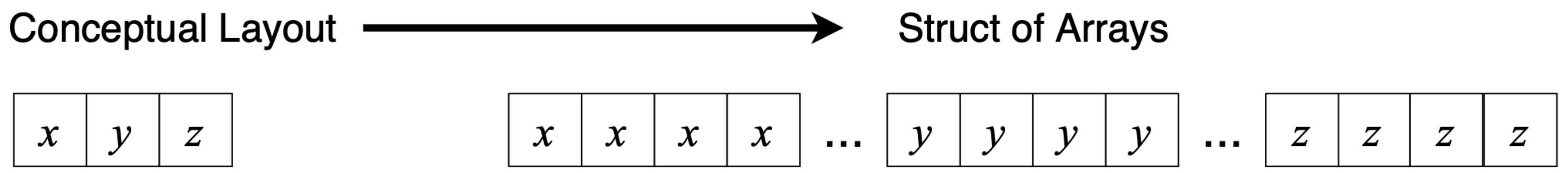
Real Time Analysis

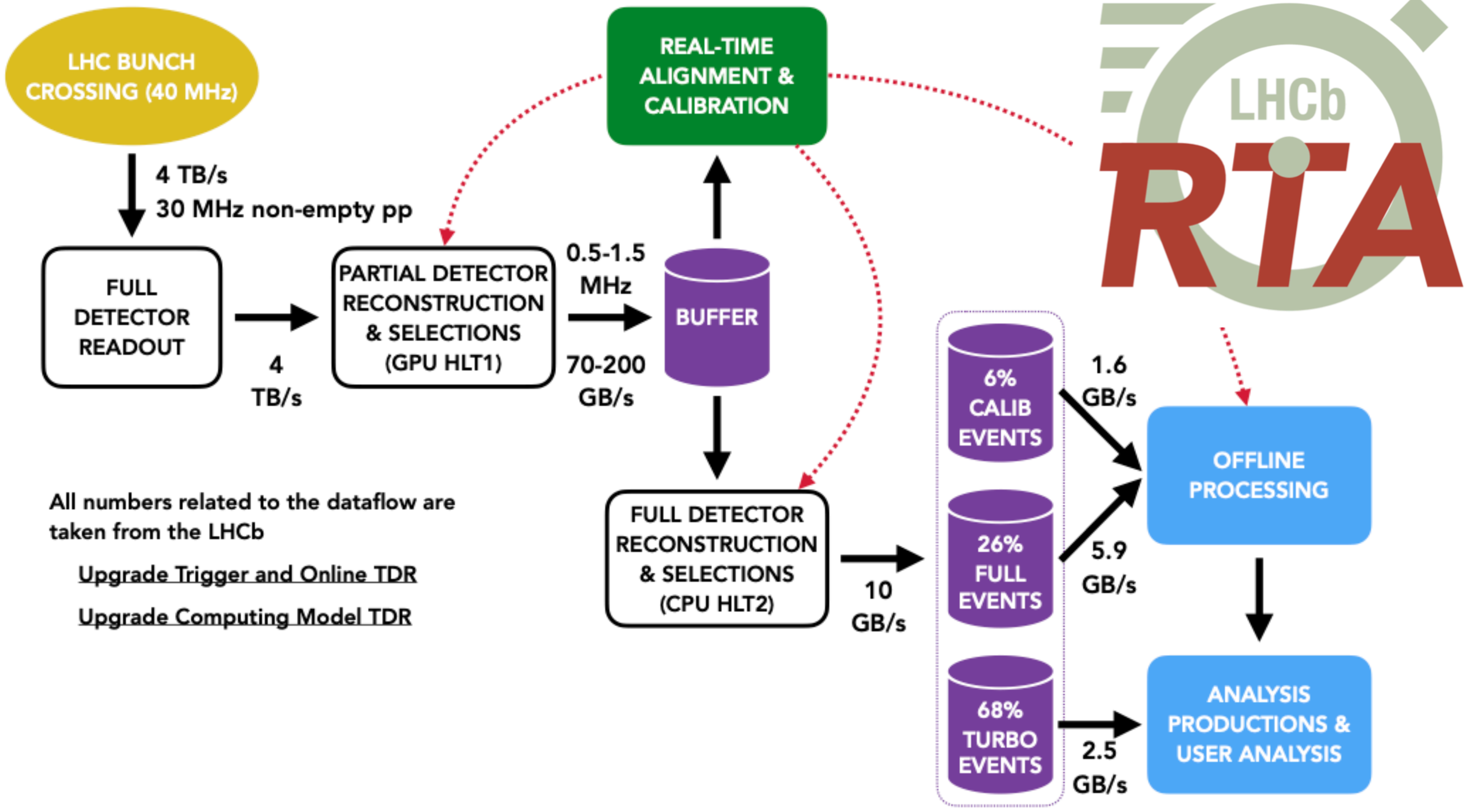
- Cannot store full events at 30MHz
- Reduce events *in the HLT* to analysis level output
- Requires fast calibration loops to ensure full offline quality in the HLT
  - No RAW data to go back to
- Optimise data layout for processing using Structure of Arrays
  - Profits from CPU SIMD instructions
  - Hide this from the end user!

Only keep the data you need for processing - smaller data is more physics per MB

Optimise the data layout for contiguous reads and parallelisation (e.g., Structures of Arrays)

SOA : Struct of Arrays - well suited for SIMD approach

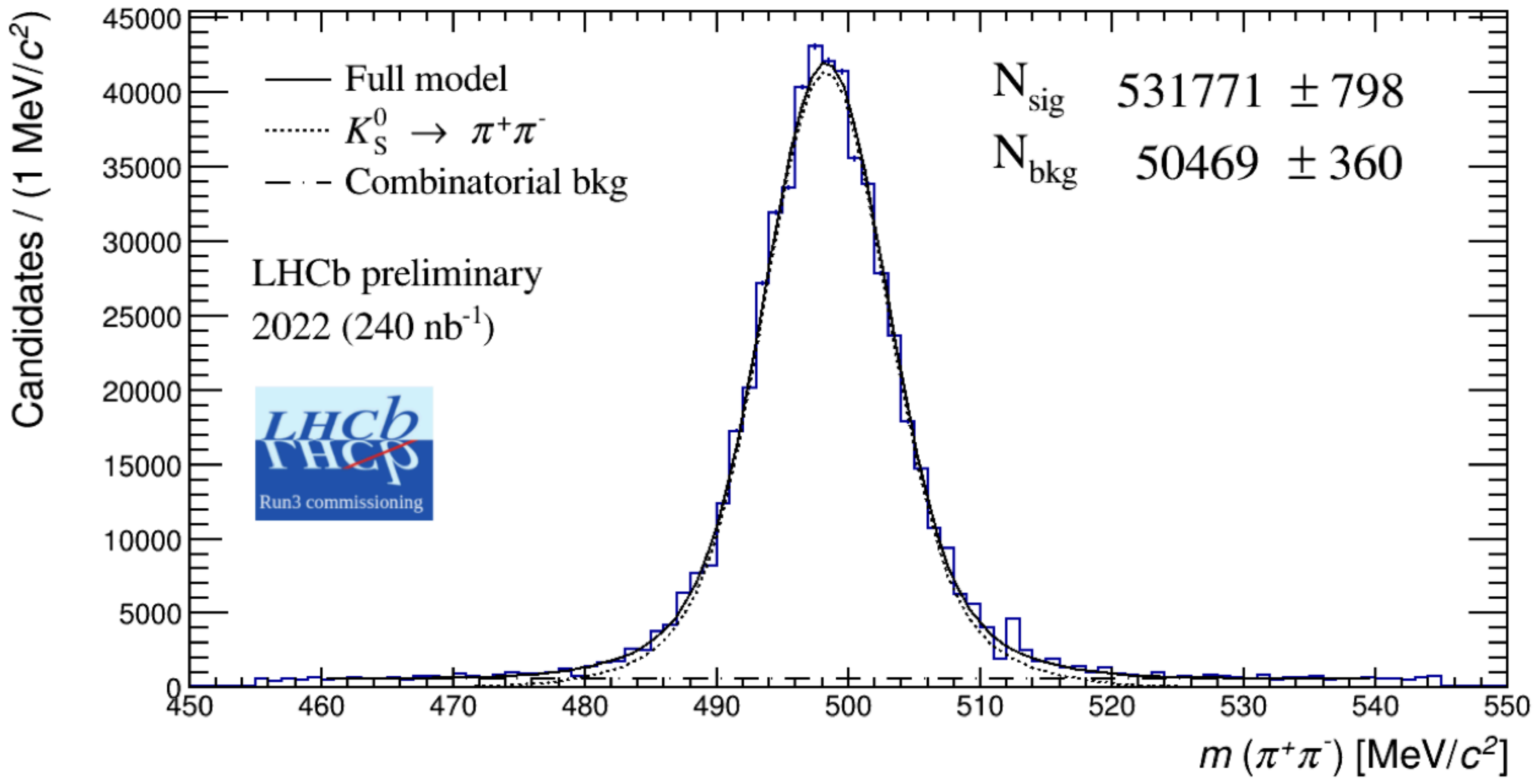



LHC BUNCH CROSSING (40 MHz)
4 TB/s
30 MHz non-empty pp
FULL DETECTOR READOUT
4 TB/s
PARTIAL DETECTOR RECONSTRUCTION & SELECTIONS (GPU HLT1)
0.5-1.5 MHz
70-200 GB/s
BUFFER
REAL-TIME ALIGNMENT & CALIBRATION
FULL DETECTOR RECONSTRUCTION & SELECTIONS (CPU HLT2)
10 GB/s
6% CALIB EVENTS
26% FULL EVENTS
68% TURBO EVENTS
1.6 GB/s
5.9 GB/s
2.5 GB/s
OFFLINE PROCESSING
ANALYSIS PRODUCTIONS & USER ANALYSIS

All numbers related to the dataflow are taken from the LHCb

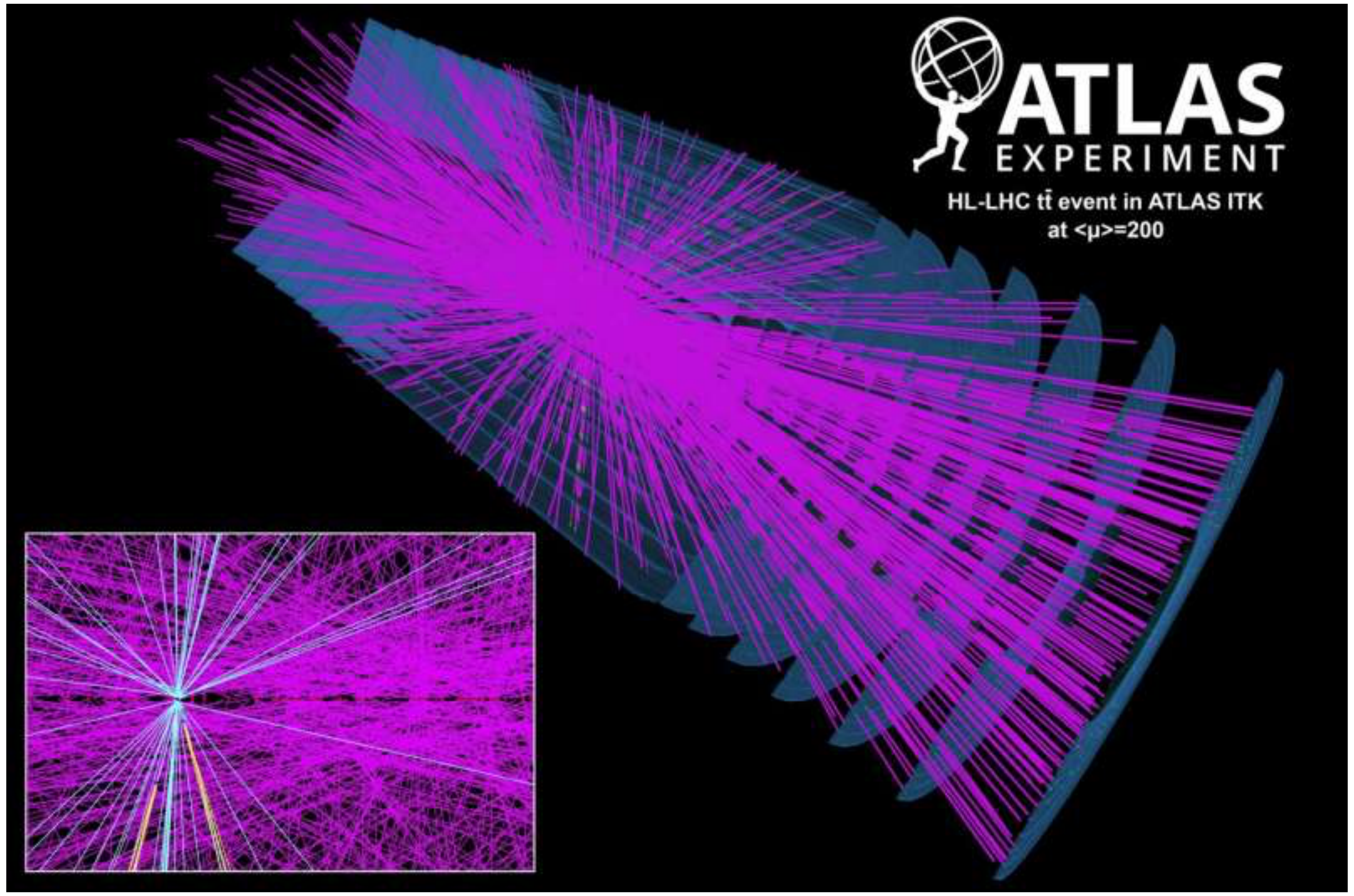
Upgrade Trigger and Online TDR

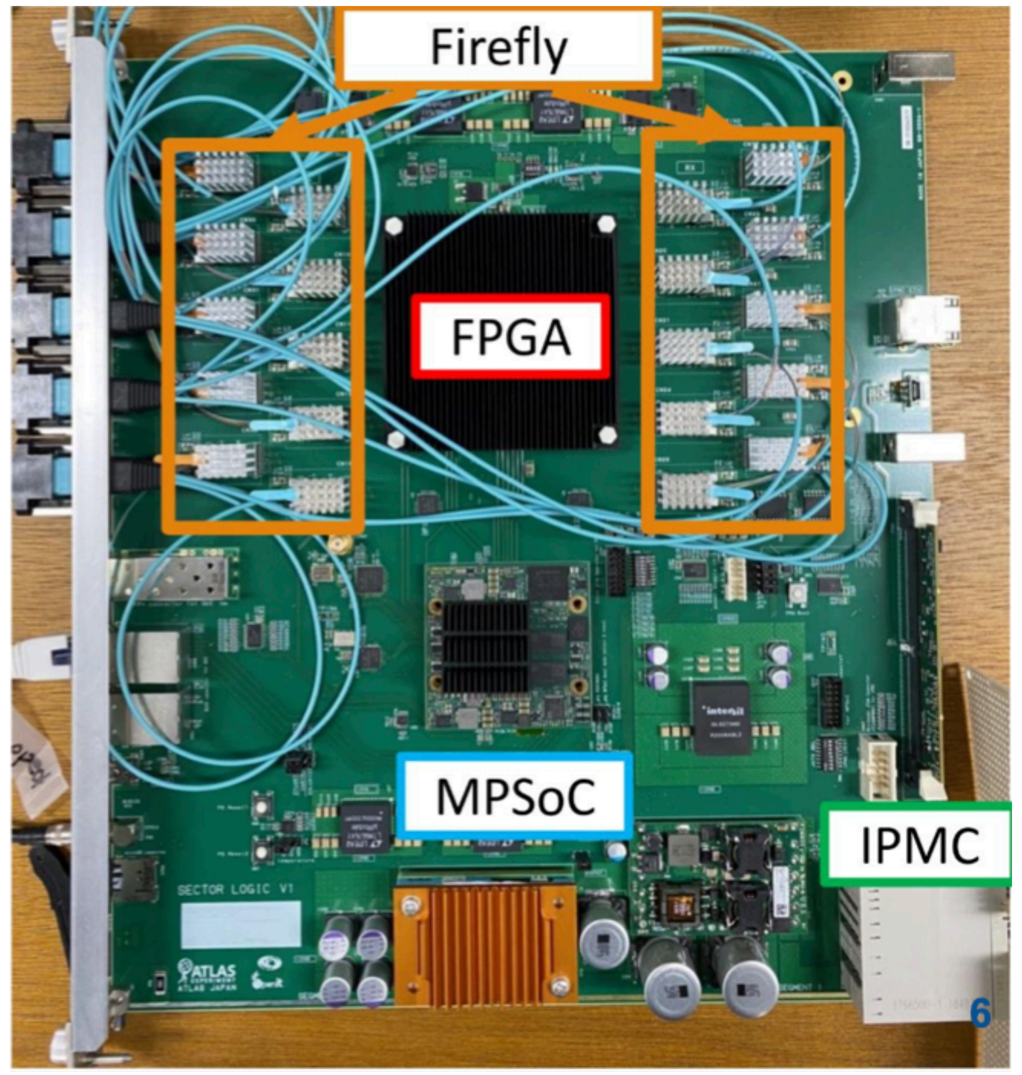
Upgrade Computing Model TDR



Physics in the Triggers

- ATLAS and CMS will be exposed to pile-up of 200 during HL-LHC
  - Event selection is far from easy with so much “noise”
  - Need access to more sophisticated event data at a lower level in the processing chain
- DAQ systems mostly based on FPGAs due to their low latency, high throughput and constant rate
- Would like to do more sophisticated physics here
  - Get access to high level information early in the data processing chain
  - But *programming these devices is not straightforward*
  - And the time/compute budget is very limited

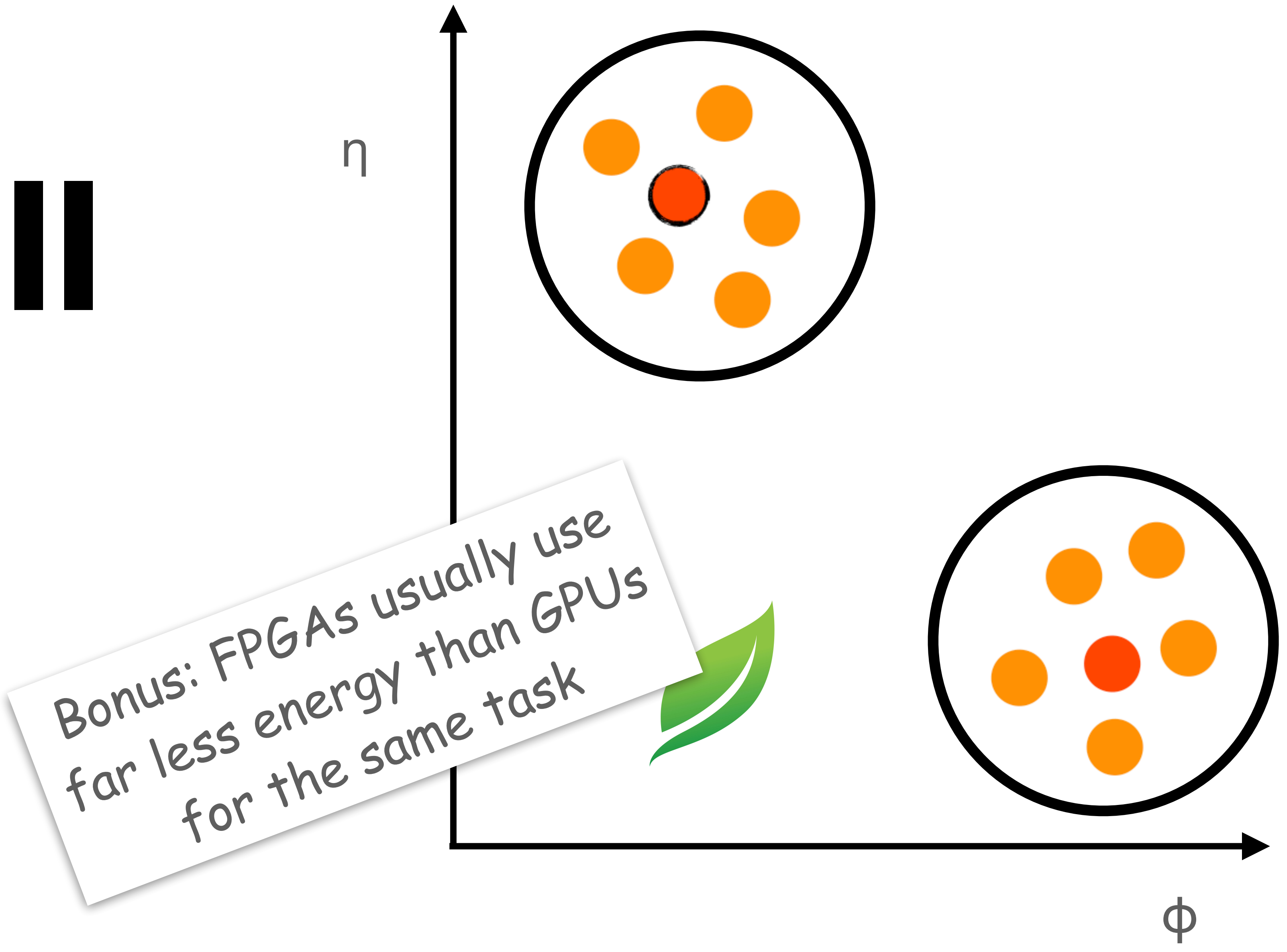




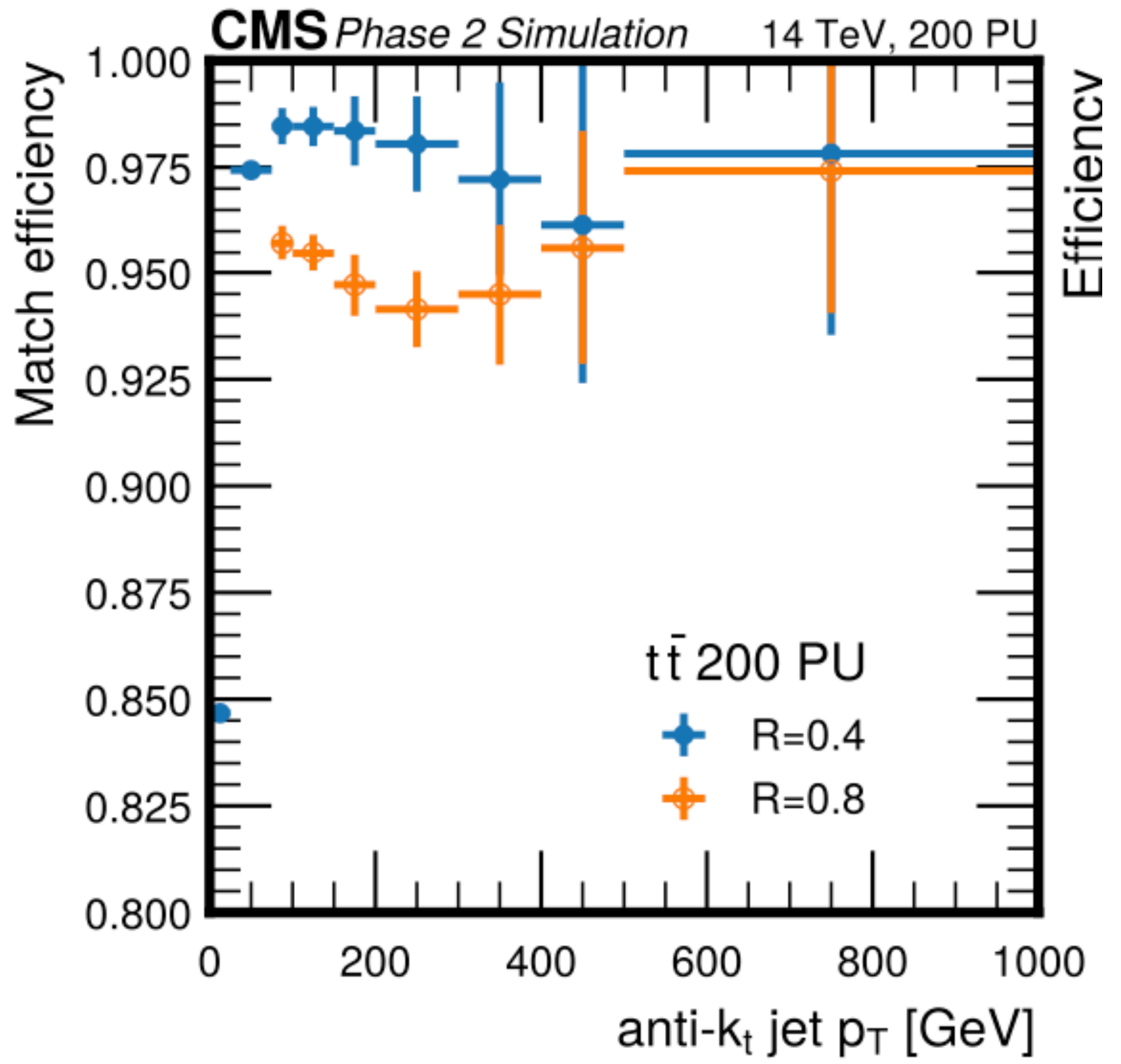
DAQ Jet Finding CMS Phase II

- Jet finding is a clusterisation problem, reassembling the decay components of a higher energy primary particle
  - As ever, data shaping is critical - assemble a flattened array of particle hits from disparate regional data
- Run a seeded cone algorithm, from the most energetic particle find the neighbours and merge into a jet, then repeat
  - Algorithm has a loop dependency - so needs to be iterative
- Use **High Level Synthesis C++** to ease programming and maintainability
- Very good matching compared to a more sophisticated offline algorithm (anti-k$_T$)
- Event processing in 744ns, pipeline processes one event per 150ns
  - 100 million jets per second!

FPGAs can be very effective and energy efficient for *some* processing steps



Bonus: FPGAs usually use far less energy than GPUs for the same task

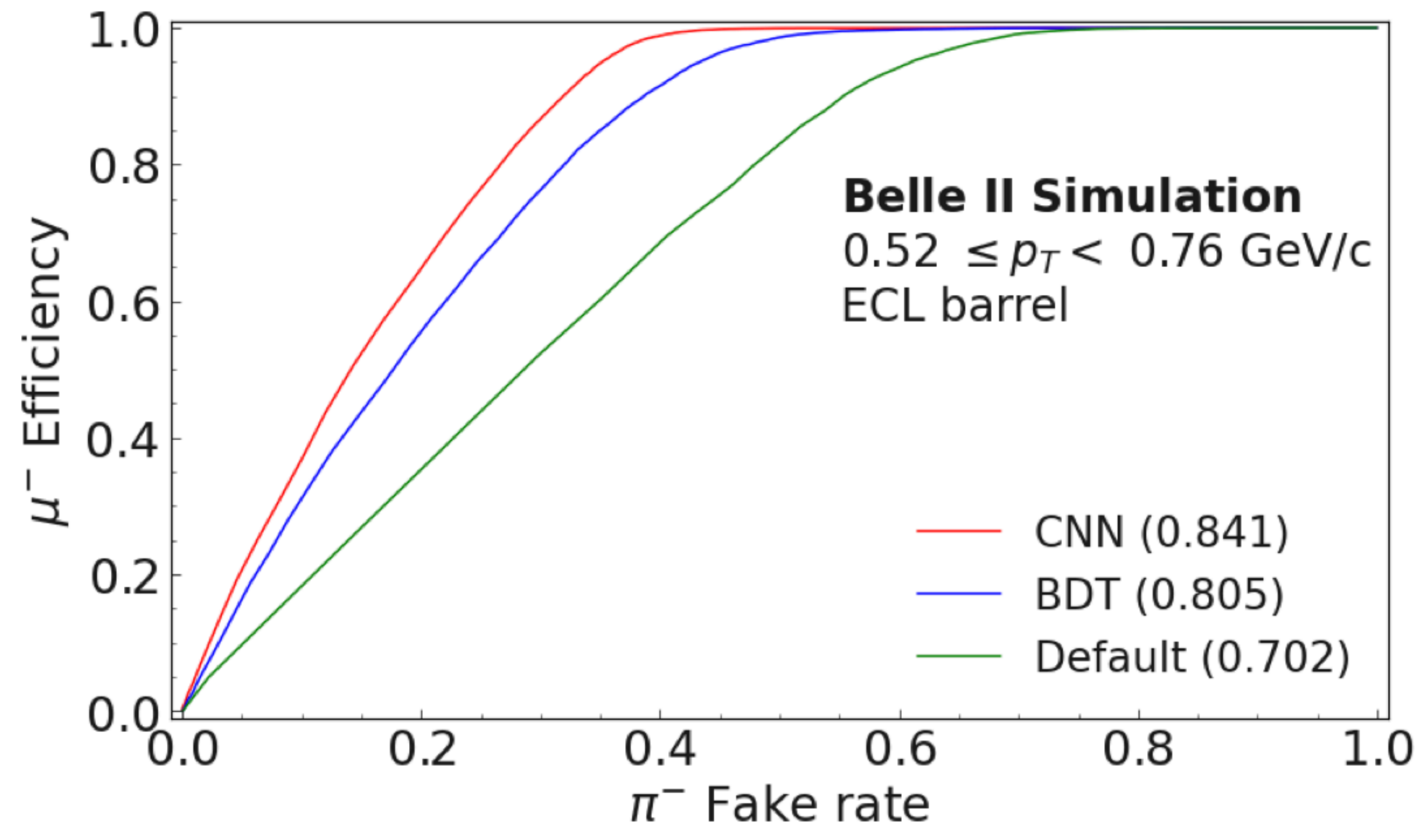


Ref: https://indico.jlab.org/event/459/contributions/11386

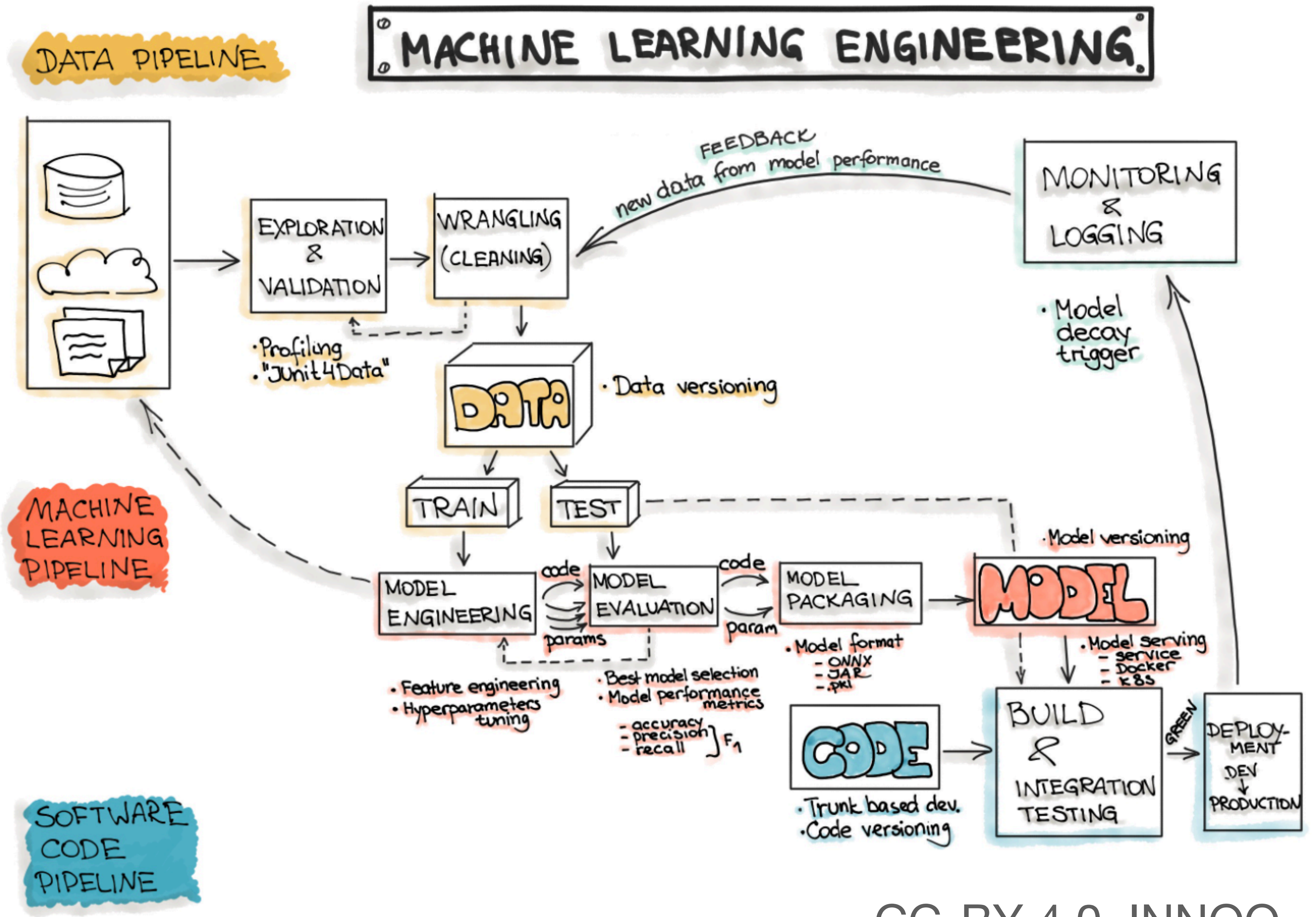
Machine Learning

- Machine learning techniques have become almost ubiquitous in HEP and in many other physics areas in the last few years
  - The current software landscape for applying these techniques has been driven by industry: TensorFlow, PyTorch, ONNX, etc.
- This has allowed sophisticated models to be developed that outperform more traditional techniques
  - e.g., particle identification from measured and reconstructed quantities
- Now widely used in analysis, jet tagging, PID, …
  - Many hot topics around uncertainty quantification, interpretability, etc.
- Training and tuning sophisticated ML models is very resource hungry
- Integration of ML inference into production workflows is non-trivial

We need to integrate machine learning workflows deeply into our software and computing



Belle II muon identification improved by the use of convolutional neural networks (CNN) vs. Boosted decision tress (BDT) and classical methods (Default) [2301.11654]



CC-BY 4.0, INNOQ

Software Meets Computing

Software Stacks

- Scientific software is not “stand alone”
  - We rely on 100s of additional software packages: generic libraries, numerical libraries, machine learning, domain specific, etc.
    - Many different build systems used (CMake, autotools, distutils, PyBuilder, etc.)
  - In addition we often build for a matrix of platforms, defined by processor, operating system and compiler
    - Aarch64, x86_86
      - Don’t forget GPU libraries for a multiplicity of different manufacturers and generations
    - RHEL X + Clones, Ubuntu, OS X (Windows we don’t do)
    - gcc11, 12, 13, … plus clang builds
      - Compiler flag variants: opt, debug, -O2/3, -March=…
- Plus we have to actually deploy the software in multiple locations, from laptops and containers to HPCs

Stack Building

- Orchestrating all of these different stack builds is a non-trivial amount of effort
  - Many different solutions grew up over the years
  - HSF Developer Tools and Packaging group studied the use cases and evaluated various tools to do this
    - We found that Spack (from LLNL) is a pretty promising tool to do this
      - There has been some interest and adoption in HEP
      - It's a good tool for librarians, but not so much for end-users
      - Excellent at building production software stacks, not so strong with the "developer story"
        - Better with a separate workflow for your experiment software
  - N.B. EESSI would be a similar alternative

Testing

- Not only does software need to be built, it’s a good idea to test it!
  - Generally relying on as much off-the-shelf infrastructure as possible
    - CMake tests, PyTest, etc.
- Hooking up the test process to your CI is the way to go
  - GitLab and GitHub have their solutions here and can integrate dedicated test resources; CDash is another way to summarise results
- N.B. multi-arch introduces an issue that results on different platforms may not be binary identical (unit of least precision, fused-multiply-add, etc.)
  - Therefore it’s important to *define carefully what an acceptable result is*
- All of this makes testing a resource intensive activity (human and machine)

Software Quality and Training

- As noted we have millions of lines of code
    - Not all of it is great…
- Modern software development workflows really help us a lot here
    - Meaningful code review was next to impossible in the SVN/CVS era
    - Automation is the key to efficiency here
        - Many code quality tests can be integrated into the CI
- But we also really need to invest in training for developers
    - HSF Training Group works with other projects and experiments to invest in training materials and running training courses
        - e.g., C++, Data Preservation, Containers, CI/CD
    - It’s very important that these efforts are recognised and rewarded!

Deployment

- For deployment, CVMFS (CernVM-FS), is certainly the most popular solution today
  - Battle tested by many communities, excellent scaling, long term support (HL-LHC lifetime and beyond)
  - The major wrinkle is site with no outbound network access, e.g., some HPCs
    - There are technical workarounds and also discussion with HPC centres about this point
    - N.B. this is also an area of common concern between different sciences, so ESCAPE, JENA can play a role here in projecting a common voice
- N.B. for *containers*, it is often better to build a container that contains the software core and a CVMFS client to use the software from `/cvmfs`
  - Greatly reduces container size and volatility
  - I guess this works for pointers to xrootd…

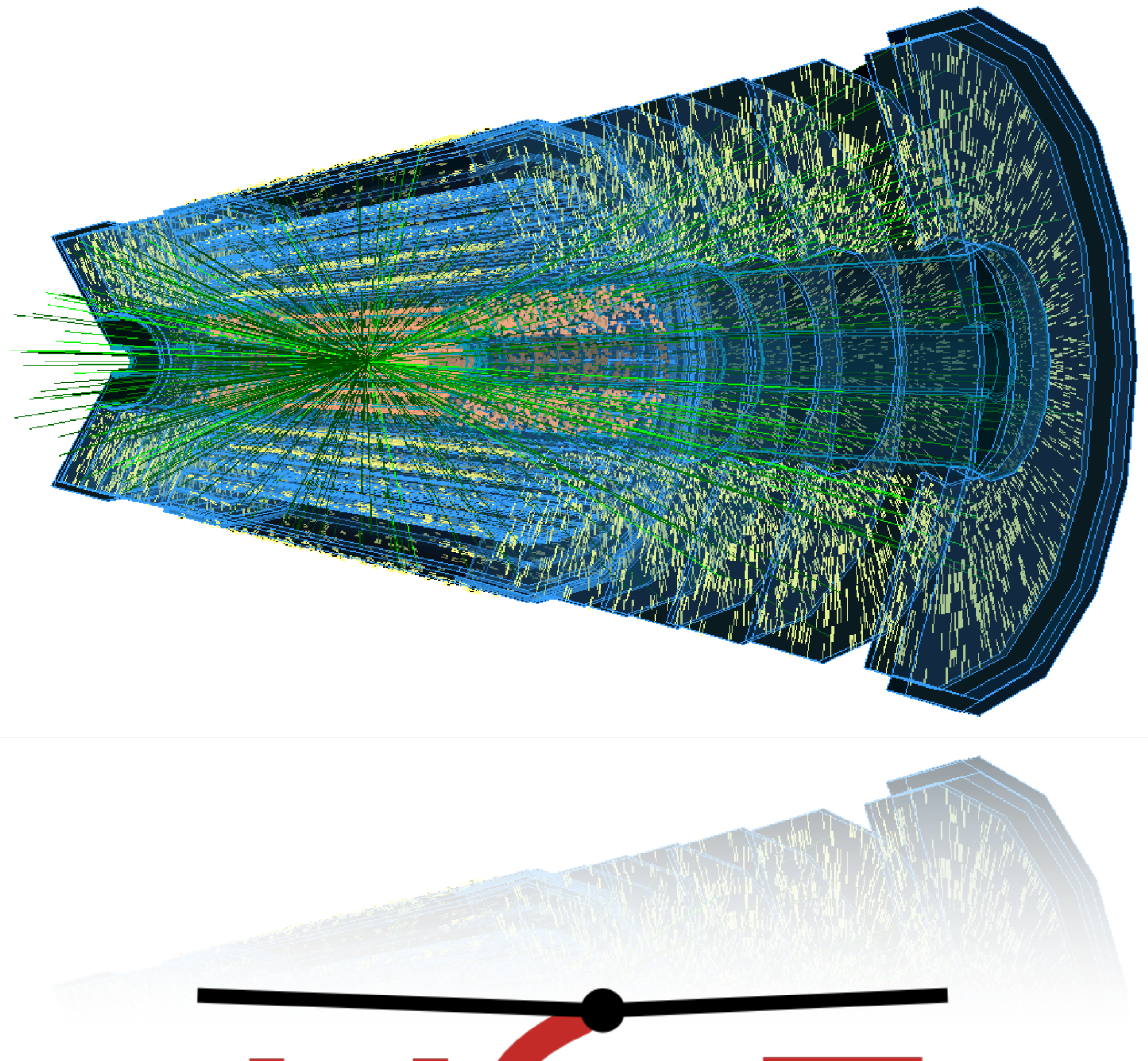
Resource Access Patterns

- LHC experiments major use of offline computing resources is in distributed high throughput mode
  - Software developers develop and test codes
  - Librarians build and deploy
  - A **production system** takes *tasks* and defines many individual batch jobs
    - These are then run across WLCG
      - DIRAC is the solution with the largest number of experiments as users
    - Results can be datasets for further batch processing or ~final physics results that become tractable on smaller scale resources (like my laptop)
- This is not the only mode, however, and not the best mode if latency matters
  - Some facilities offer *interactive multi-node setups*, e.g., Jupyter
    - Great for certain development phases
    - Ease considerably software setup issues
    - Friction can happen at breakpoints from local to cluster to batch
  - Significant interest in tying these types of access to high performance storage systems, allowing rapid turn around on very large amounts of event data (so called *analysis facilities*)

Summary and Conclusions

Conclusions

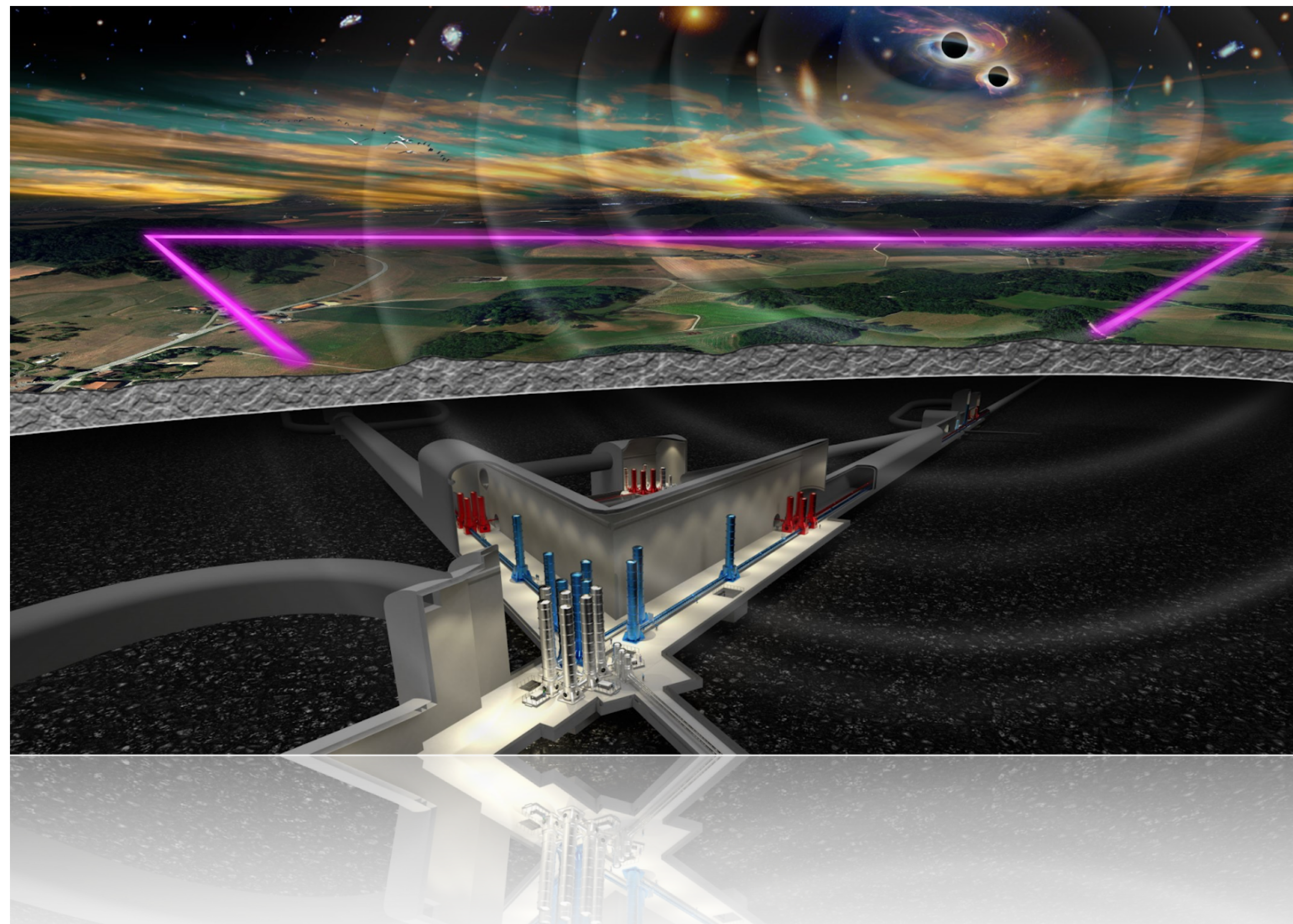
- Exciting physics programs ahead in many areas
  - These bring high data rates and data volumes
- Software and computing are *mission critical areas*
- The use of **compute accelerators** enables us to keep up with high data processing rates and growing computing requirements
  - However, not easy to translate all parts of the workflow (and CPUs do not go away)
- Optimal data handling is vital to achieve necessary throughput
- **Machine learning** brings benefits in many areas, still exploring
  - Not yet clear what the final impact on resources and workflows will be
- There is a large contact surface between software and distributed computing
  - Many issues need to be addressed reliably build and deploy well tested software and to interface with different resource flavours
  - Developer and analyst time is very precious - training and automation help people to be efficient in doing science



HSF

HEP Software Foundation

Community Matters!



Backup

HEP Software Foundation

HEP Software Foundation

- Software and computing challenges are faced across HEP experiments **and in other science areas**
  - HEP software must evolve to meet these challenges and exploit all expertise
- New experiments should not be starting from scratch, but building on best-of-breed
  - The role of the HSF, started in 2015, and now well established in the field, is to facilitate cooperation and common efforts in software and computing across HEP in general
  - Our philosophy is bottom up, a.k.a. do-ocracy
- Organises community meetings in important areas for the field (simulation, reconstruction, etc.)
- Has had a substantial impact in many areas
  - PyHEP - Data science tools and Python for HEP (PyHEP2023)
  - Computational aspects of event generators (N(N)LO workshop)
  - Training (our training centre)
- Incubation of new ideas and trends: this year we started organising work on Julia in HEP (workshop)

General mailing list:
hsf-forum@googlegroups.com

Coordination team:
hsf-coordination@googlegroups.com

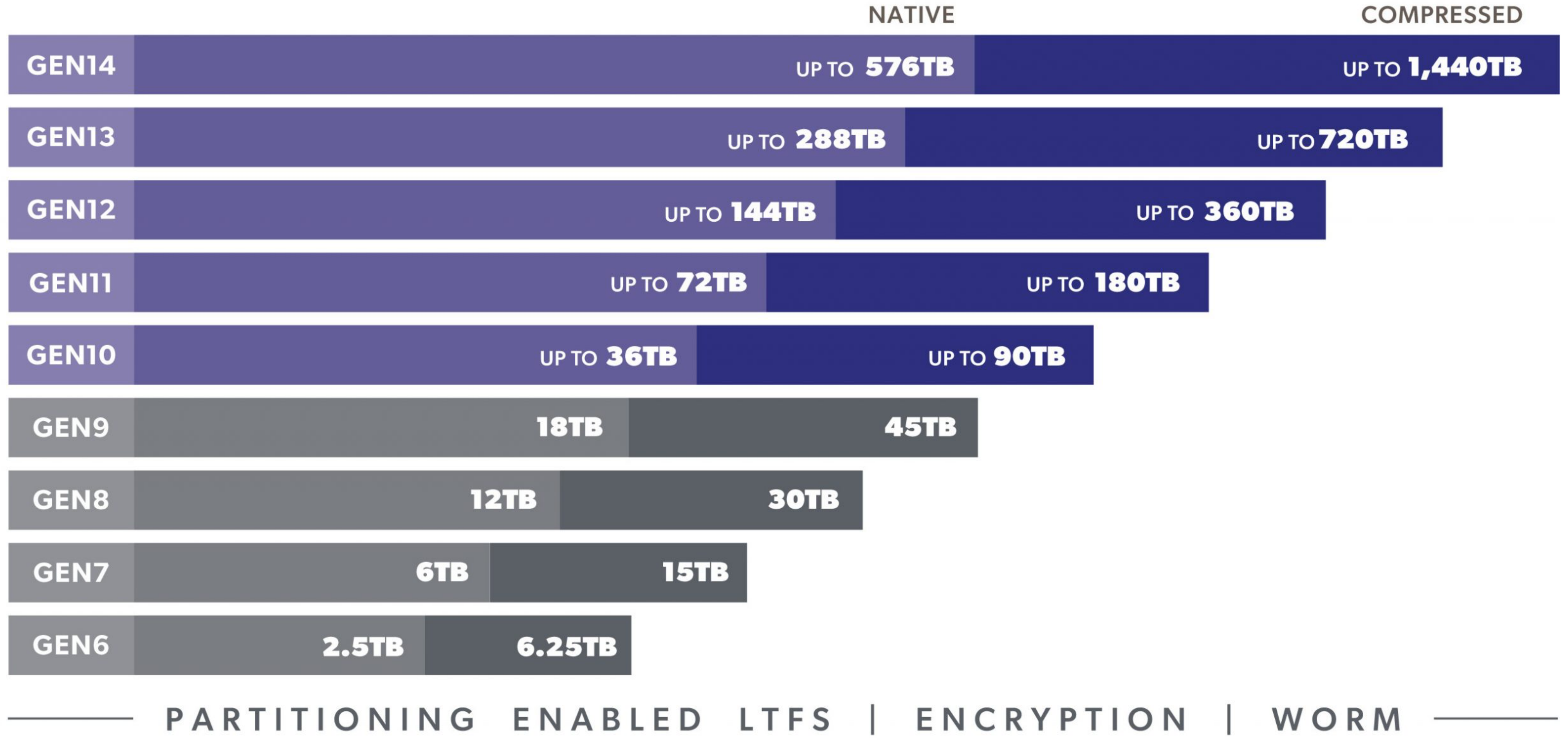
Always happy to have useful collaborations with other fields!

Storage

- WLCG spends more on storage than compute
  - This is unlikely to change with HL-LHC
- Hard drives continue to store the majority of the world's data today
  - Zettabytes of HDD capacity sold in recent years
- SSD prices do keep falling, this year quite fast
  - This makes SSDs more attractive in the data centre (more reliable and lower power costs than HDD)
- Tape market is steady for volume, with continued technology improvements
  - Cost per TB will remain the lowest for some time
- *Probably we are ok here, but exact technology mix for the future hard to predict*

LTO ULTRIUM ROADMAP

Addressing your storage needs

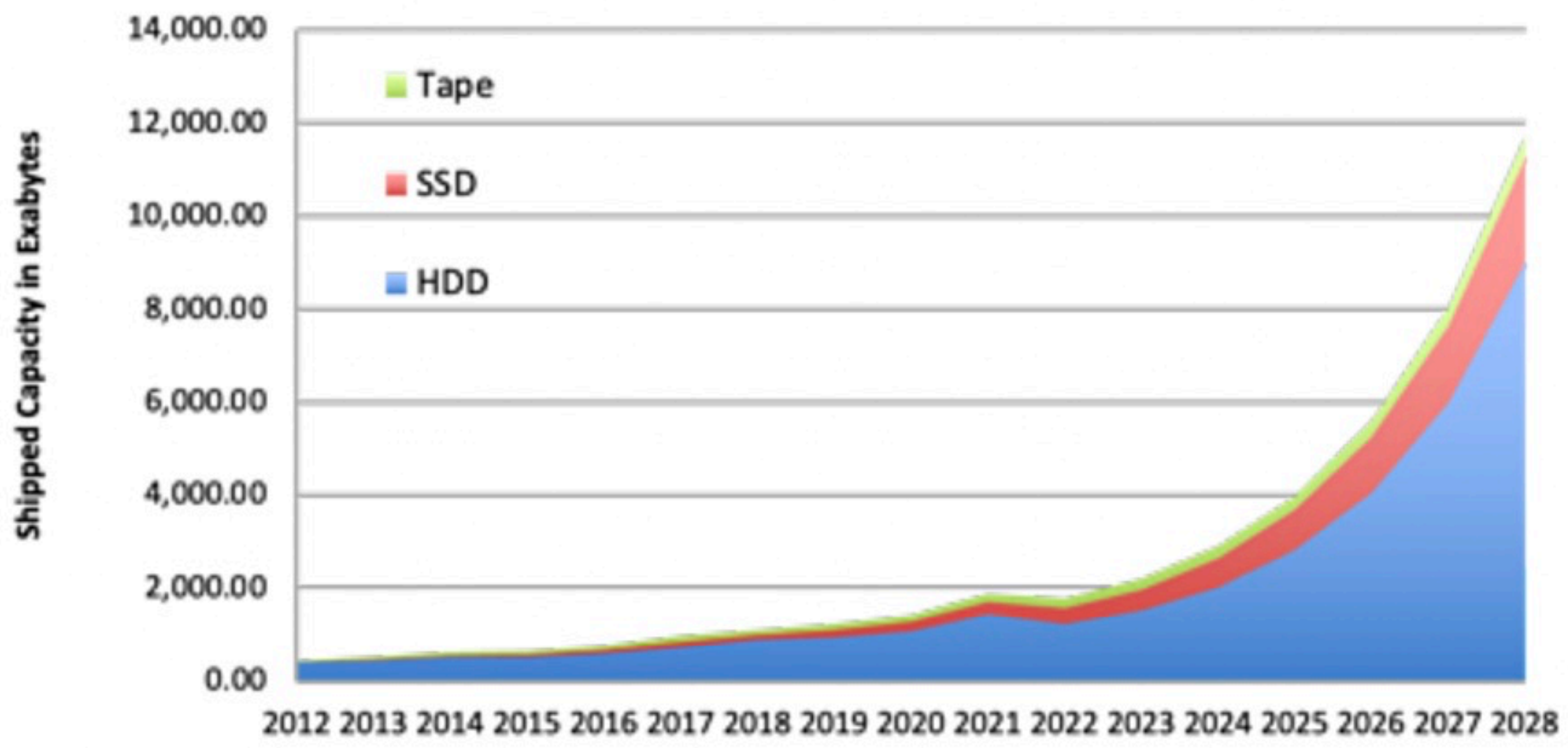


NOTE: Compressed capacities assume 2.5:1 compression (achieved with larger compression history buffer).

SOURCE: The LTO Program. The LTO Ultrium roadmap is subject to change without notice and represents goals and objectives only. Linear Tape-Open LTO, the LTO logo, Ultrium and the Ultrium logo are registered trademarks of Hewlett Packard Enterprise Company, International Business Machines Corporation and Quantum Corporation in the US and other countries. Please contact your supplier/manufacturer for more information.

Hewlett Packard Enterprise Company, International Business Machines Corporation and Quantum Corporation collaborate and support technology specifications, licensing, and promotions of LTO Ultrium products.

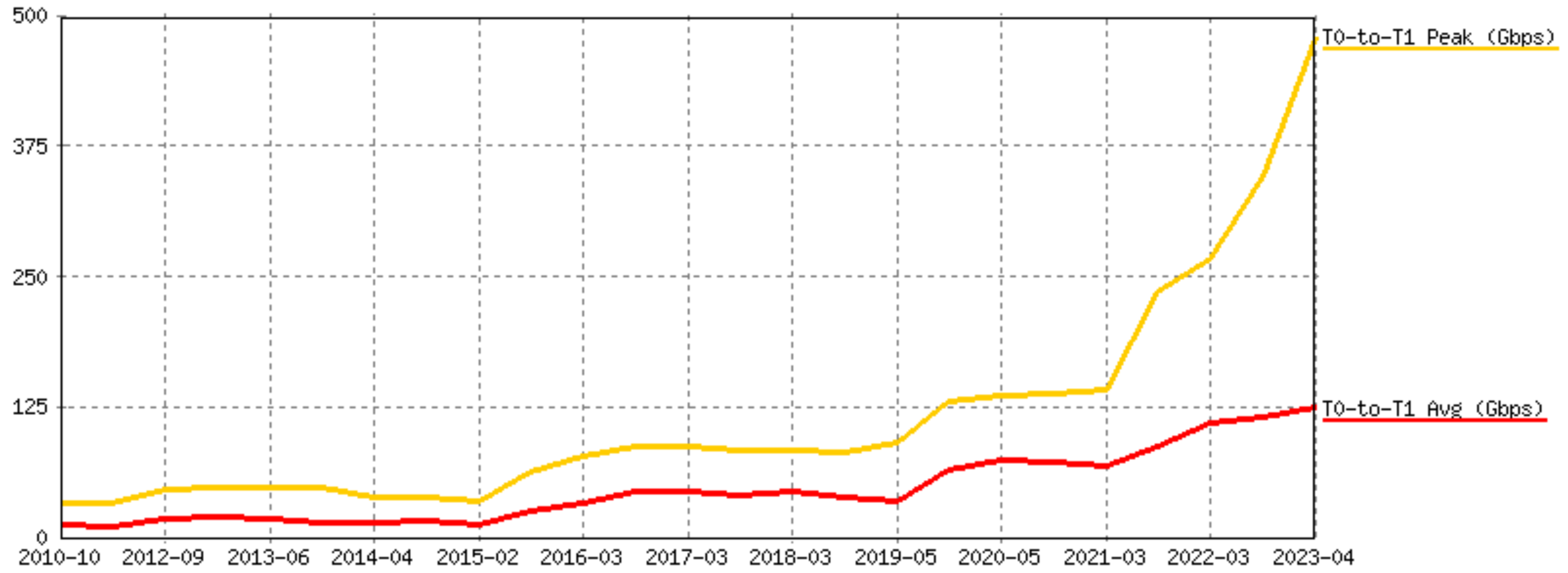
FIGURE 11. CAPACITY SHIPMENTS FOR LTO TAPE, SSDS AND HDDS



August 2023 Digital Storage Technology newsletter

Network

- Network has been an outstanding infrastructure success for LHC
  - Capacity and growth far exceeded initial planning
  - This has enabled a substantially different computing model from that first envisaged for LHC
- No sign that this will really slow down
- 400Gb circuits now being commissioned for academic networks
- *Network is a resource that we can have quite some confidence in going forwards*



LHCONE Network Transfers over 10 Years

Data Carried

7PB

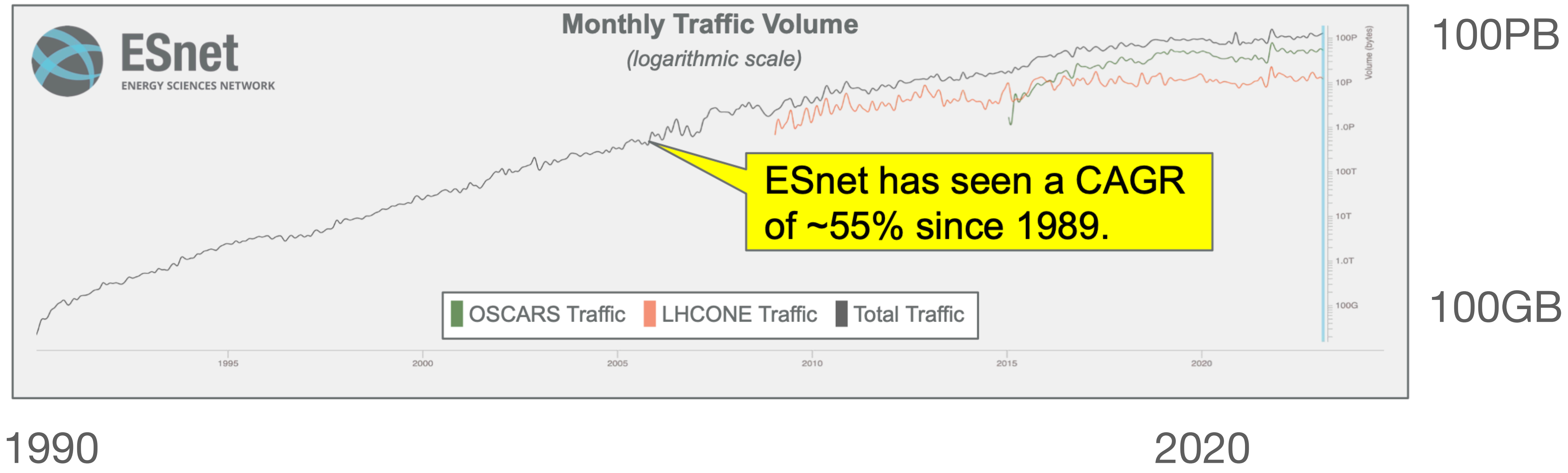
7 Petabytes of data carried per day

Traffic Growth

+30%

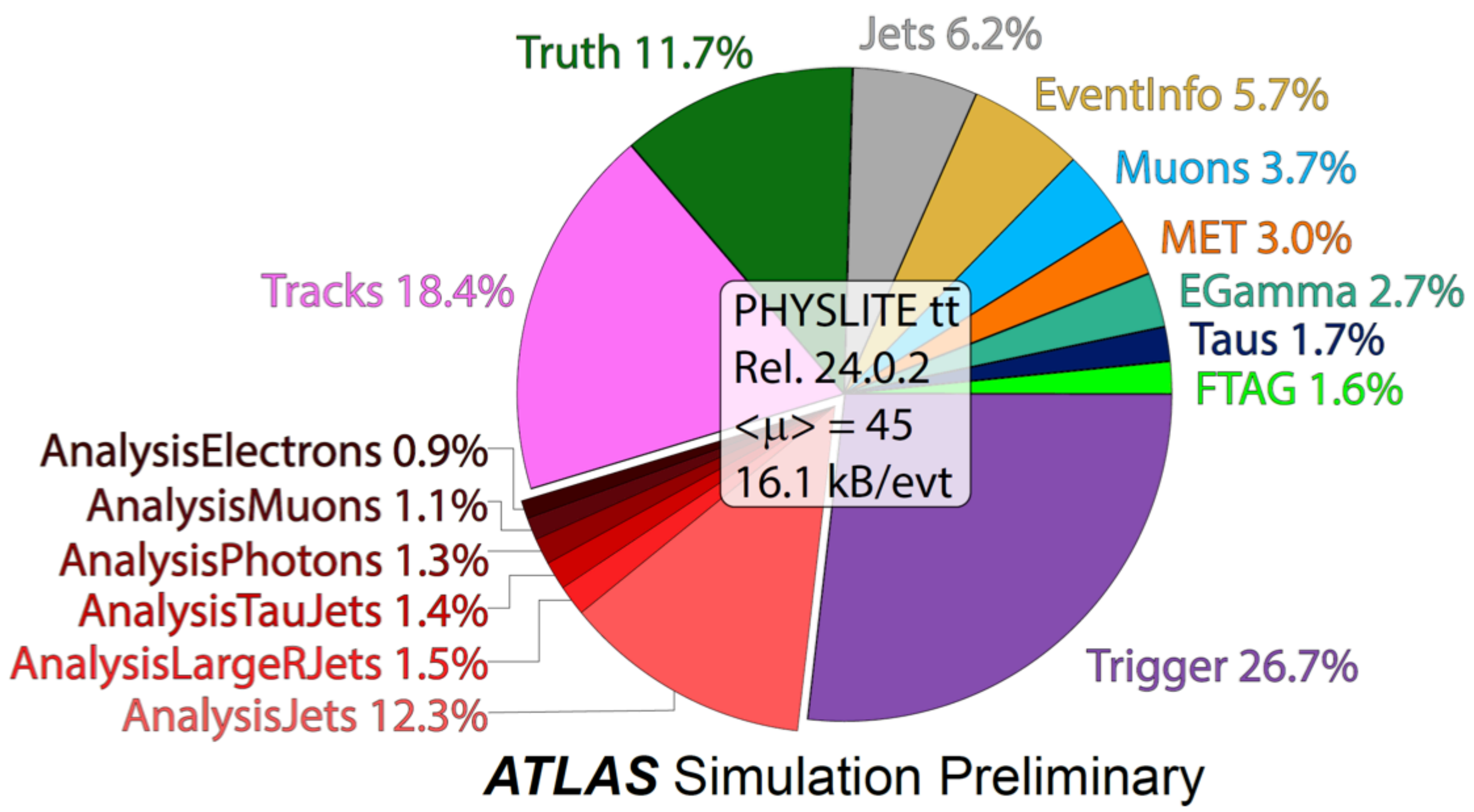
Average annual increase in network traffic over last five years

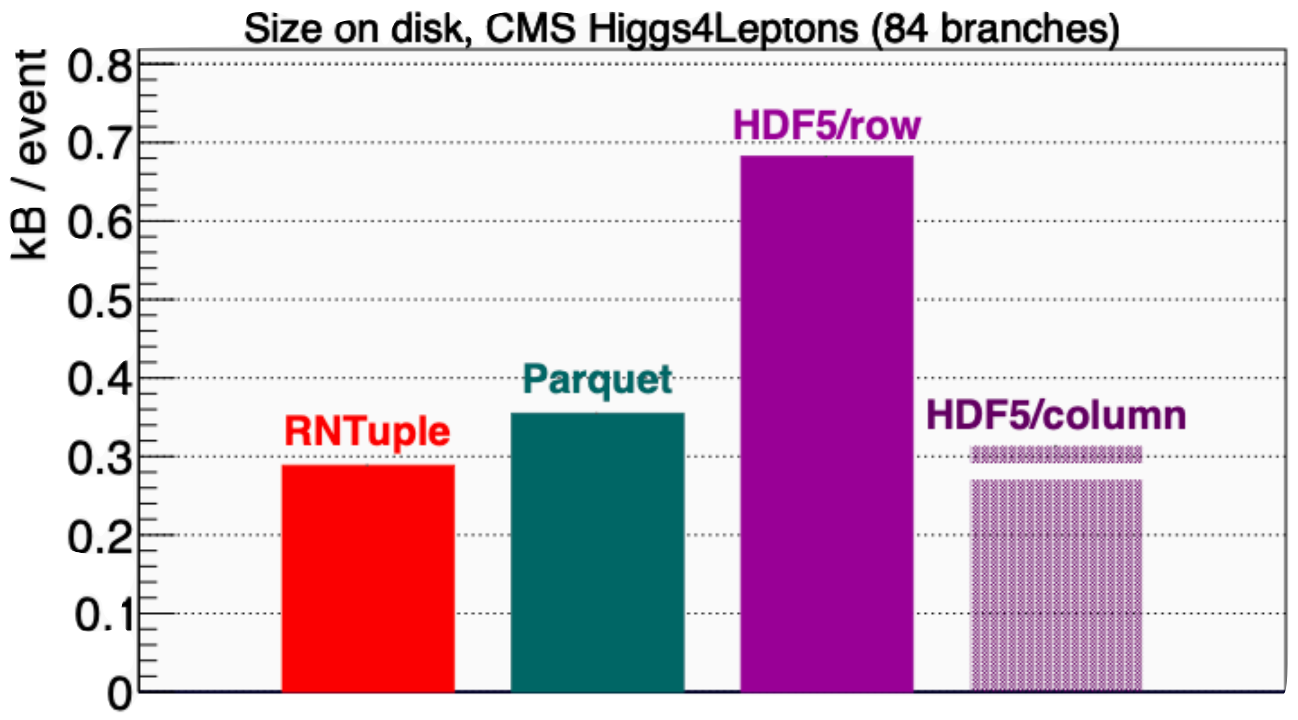
GEANT European Academic Network carrying 7PB day, 30% year on year growth

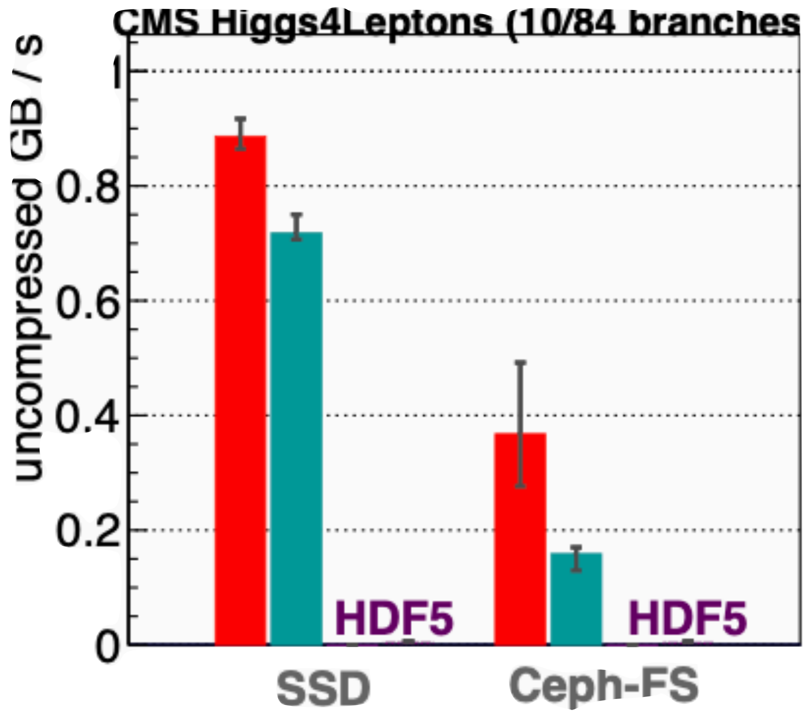


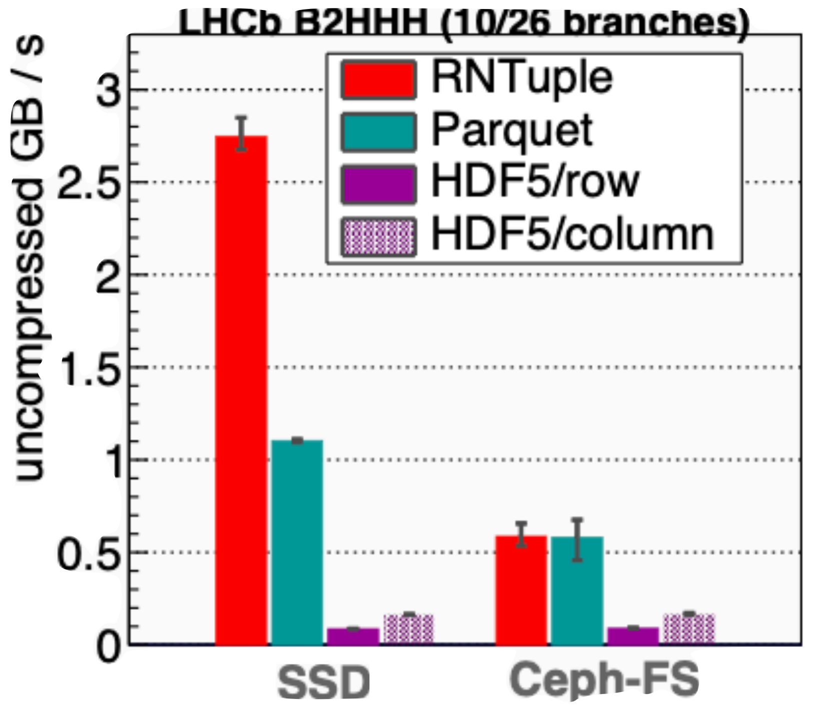
ESnet has seen a CAGR of ~55% since 1989.

Small is Beautiful







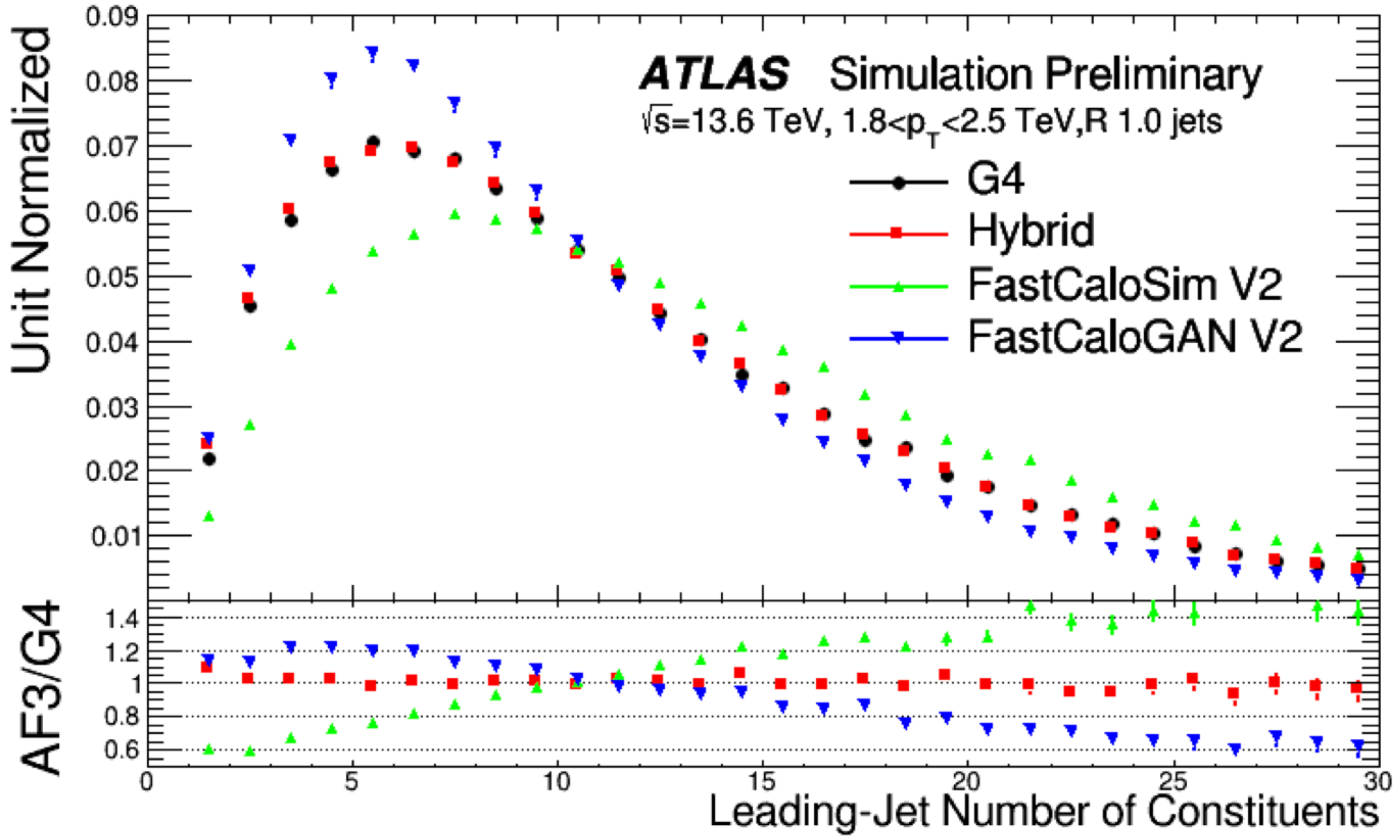


- Big detectors = big data!
  - e.g., ATLAS Analysis Object Data is 300-500kB/event
  - At 10kHz trigger rate this can't even fit on disk anymore!
    - Data carousel used to **progressively stage data and process from *tape***
- Need to aggressively reduce data volumes to manageable levels
  - New data format (DAOD_PHYSLITE): pre-calibrated and suitable for around 80% of analysis use cases
  - Target 10-12kB average per event - x40 reduction from initial AOD
- New underlying data format, ROOT RNTuple, is best in class for size on disk and for read speeds
  - Similar technology to Parquet, but more optimal for HEP data
- **Smaller data formats mean more physics per MB and per second**

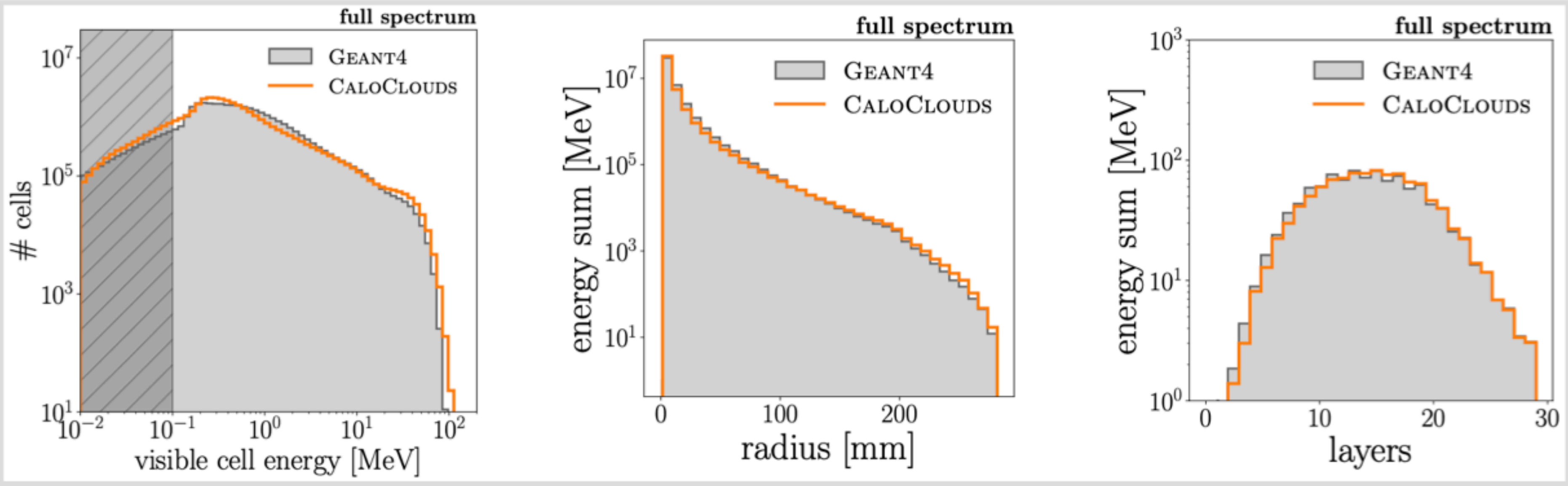
Ref: https://indico.jlab.org/event/459/contributions/11586/
Ref: https://indico.cern.ch/event/1294815/

ML Application Area: Simulation

- Simulation in HEP is critical to understand the detector and analyse data
    - It is also computationally *very expensive* to do in Geant4
        - Unfeasible to use this for all simulation even in Run 3
    - ML generated events can be a good way to replace (parts of) full simulation or enhance parametric simulation
        - They are usually fast - can they be sufficiently accurate?
- Some nice results from ATLAS now using GANs in their fast simulation (CaloGAN)
    - Divide and conquer - 100 slices in η and separate networks for γ, π±, e, p
- Studies beginning in new techniques that are popular in industry, viz. transformers (DALL-E, ChatGPT) and diffusion models
    - Encouraging early results on ILD Ecal simulation (but much slower than other techniques)



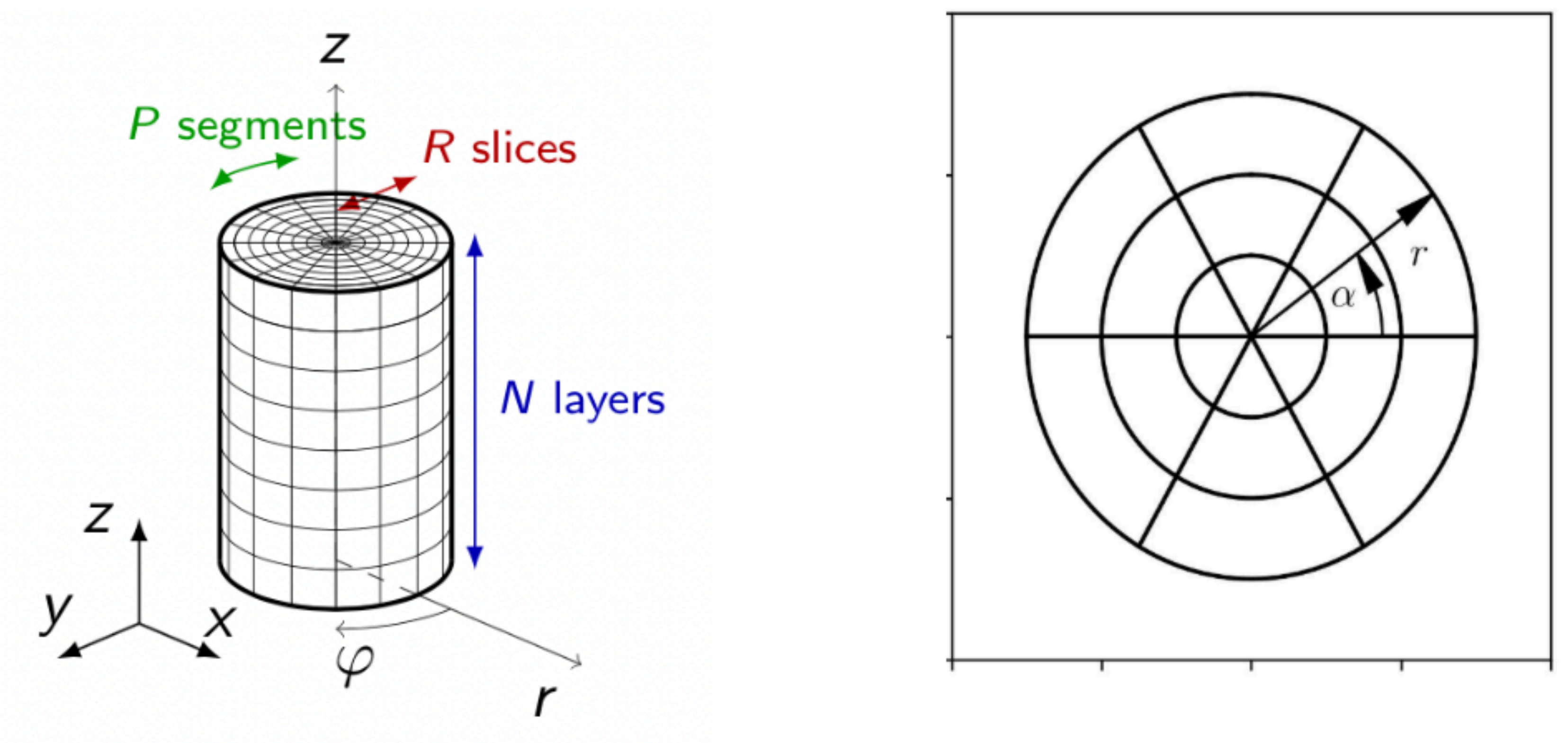
AtlFast3 - with GAN generated responses [2109.02551]



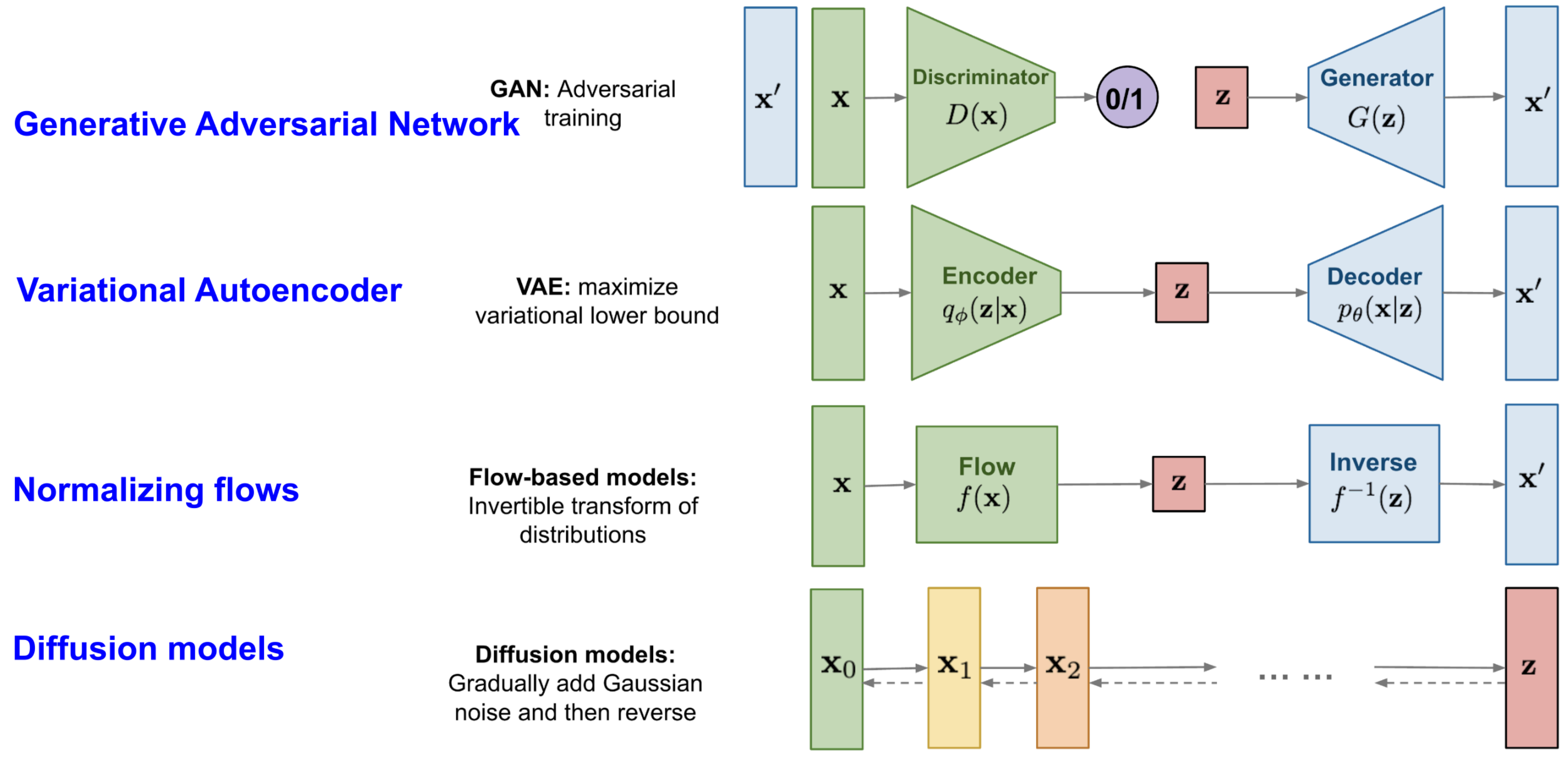
Diffusion model results for ILC EM calorimeter

ML Standard Candles

- One issue with ML techniques (and there are a lot of them!) is how to compare between different detectors and experiments
- A way to overcome this is to have a standard set of data, which is meaningful, but experiment neutral
- For this kind of generative simulation the initiative is the CaloChallenge datasets
    - Three datasets of fully simulated showers of different complexity
- Many models investigated: VAE, GAN, Flow, Diffusion
- Covering many interesting choices of data representation, scaling and conditioning, model stages, etc.



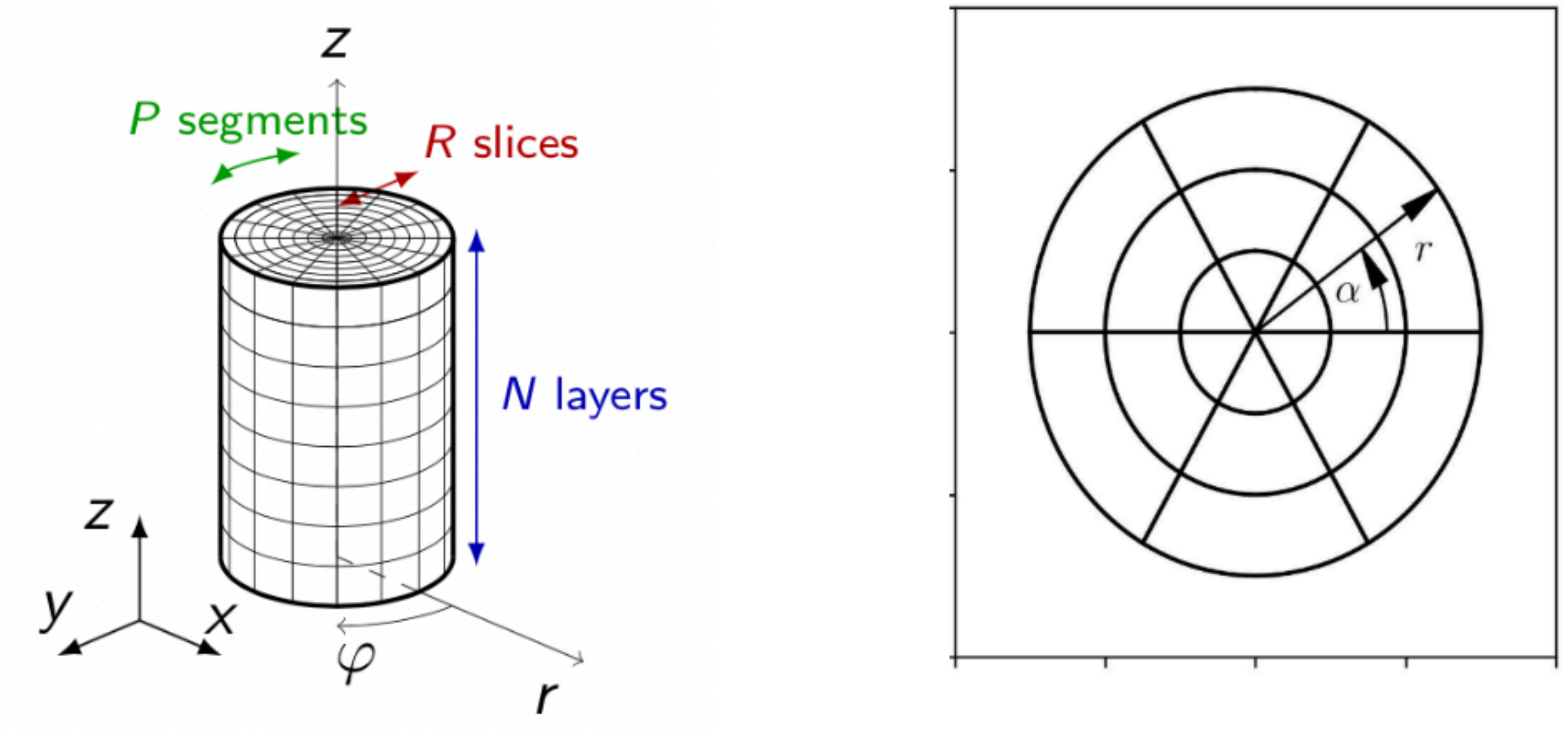
Geometrical layout of Calo Challenge datasets
#1 - ATLAS-like γ, π showers (368D, 533D)
#2 - SiW sampling calorimeter (6480D)
#3 - SiW high granularity (40500D)


Generative Adversarial Network
GAN: Adversarial training
$\mathbf{x}'$
$\mathbf{x}$
Discriminator $D(\mathbf{x})$
0/1
$\mathbf{z}$
Generator $G(\mathbf{z})$
$\mathbf{x}'$
Variational Autoencoder
VAE: maximize variational lower bound
$\mathbf{x}$
Encoder $q_\phi(\mathbf{z}|\mathbf{x})$
$\mathbf{z}$
Decoder $p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$
Normalizing flows
Flow-based models: Invertible transform of distributions
$\mathbf{x}$
Flow $f(\mathbf{x})$
$\mathbf{z}$
Inverse $f^{-1}(\mathbf{z})$
$\mathbf{x}'$
Diffusion models
Diffusion models: Gradually add Gaussian noise and then reverse
$\mathbf{x}_0$
$\mathbf{x}_1$
$\mathbf{x}_2$
... ...
$\mathbf{z}$

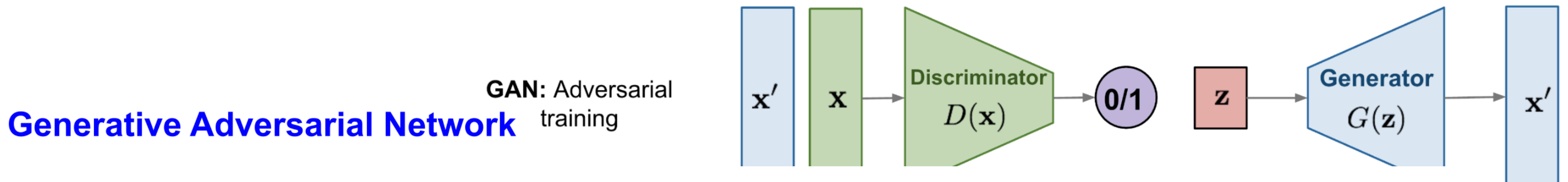
Sketch of different model architectures

ML Standard Candles

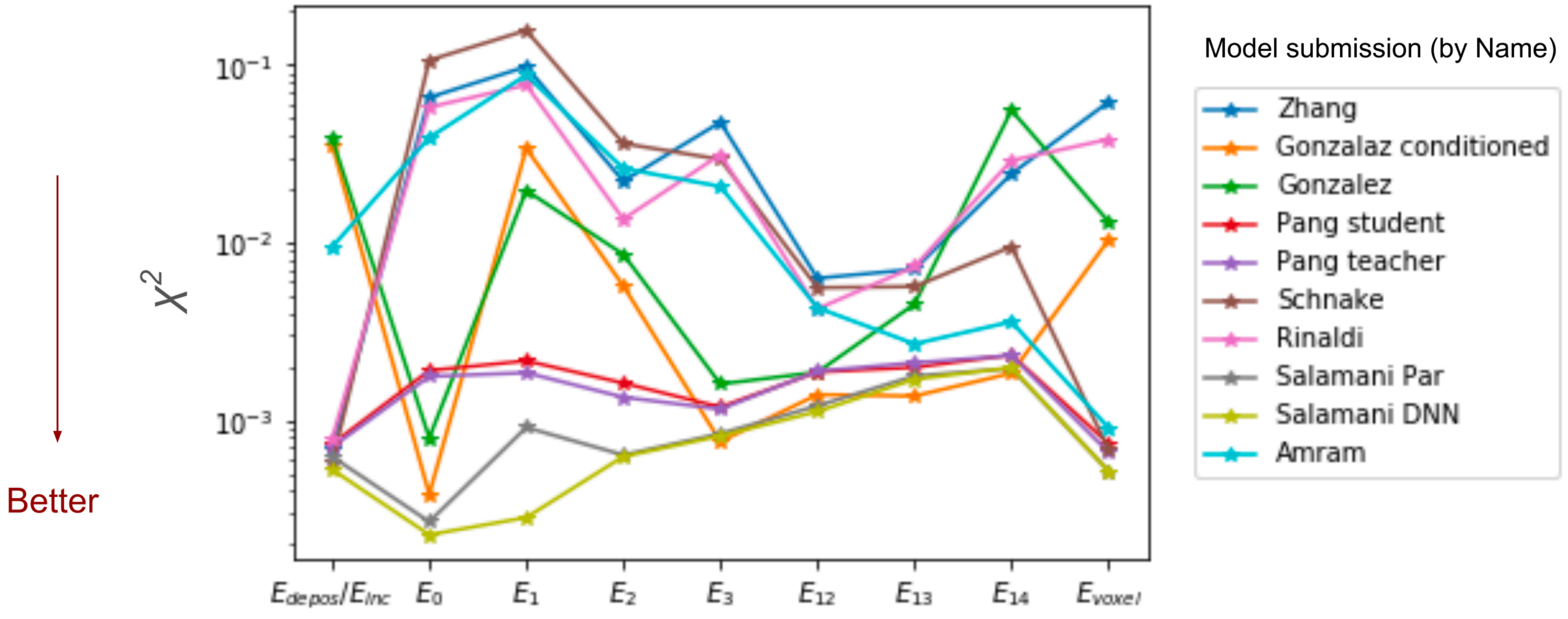
- One issue with ML techniques (and there are a lot of them!) is how to compare between different detectors and experiments
- A way to overcome this is to have a standard set of data, which is meaningful, but experiment neutral
- For this kind of generative simulation the initiative is the CaloChallenge datasets
  - Three datasets of fully simulated showers of different complexity
- Many models investigated: VAE, GAN, Flow, Diffusion
- Covering many interesting choices of data representation, scaling and conditioning, model stages, etc.



Geometrical layout of Calo Challenge datasets
#1 - ATLAS-like γ, π showers (368D, 533D)
#2 - SiW sampling calorimeter (6480D)
#3 - SiW high granularity (40500D)



Looking at histogram χ^2 of Etot/Einc, Elayer$_i$ (energy per calorimeter layer) , Evoxel



Permodel results for total, layer and voxel energy matching