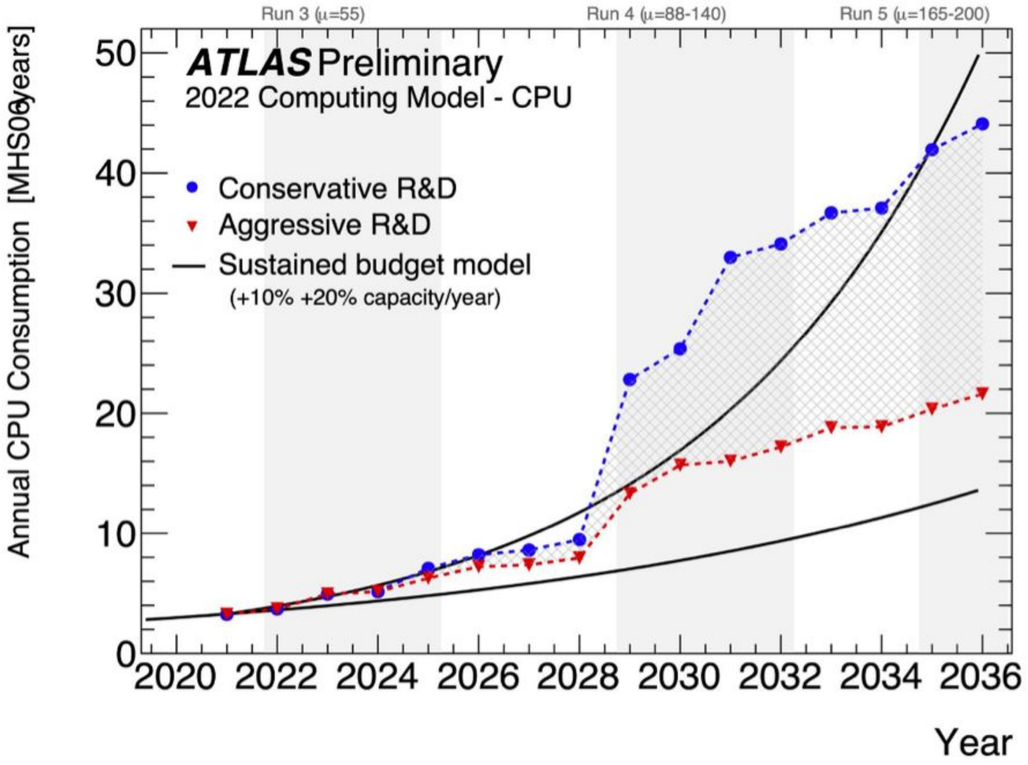
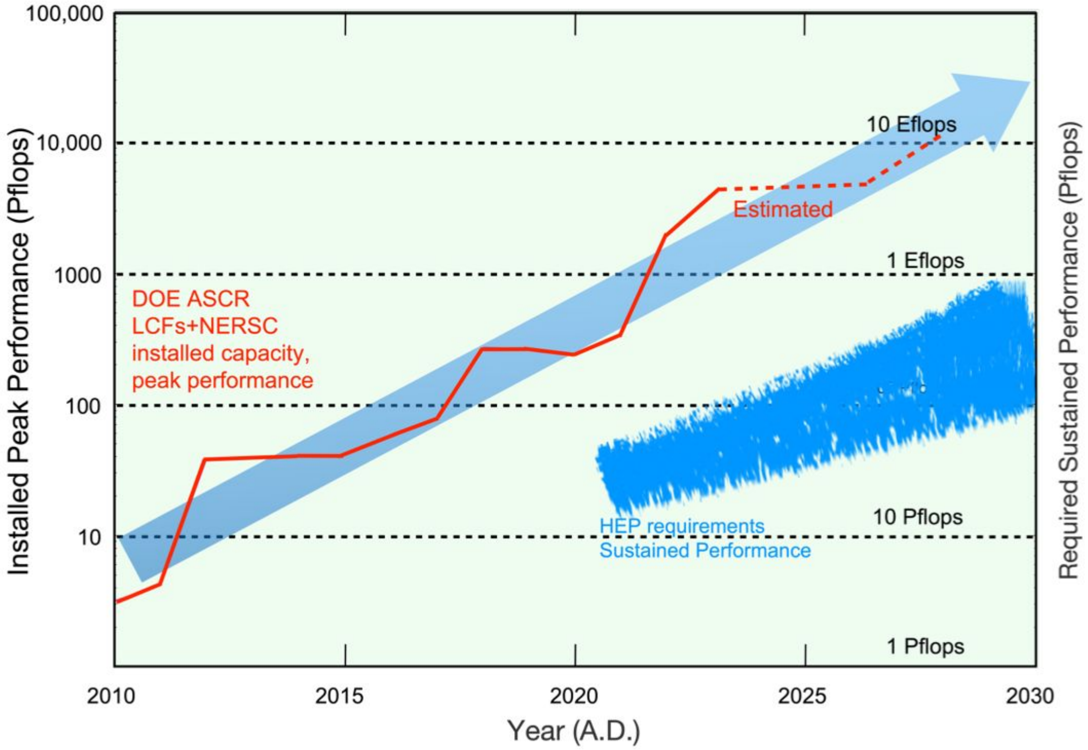
Computing Resources for Future HEP Experiments





CHALLENGE:
Increased computing requirements over coming years.

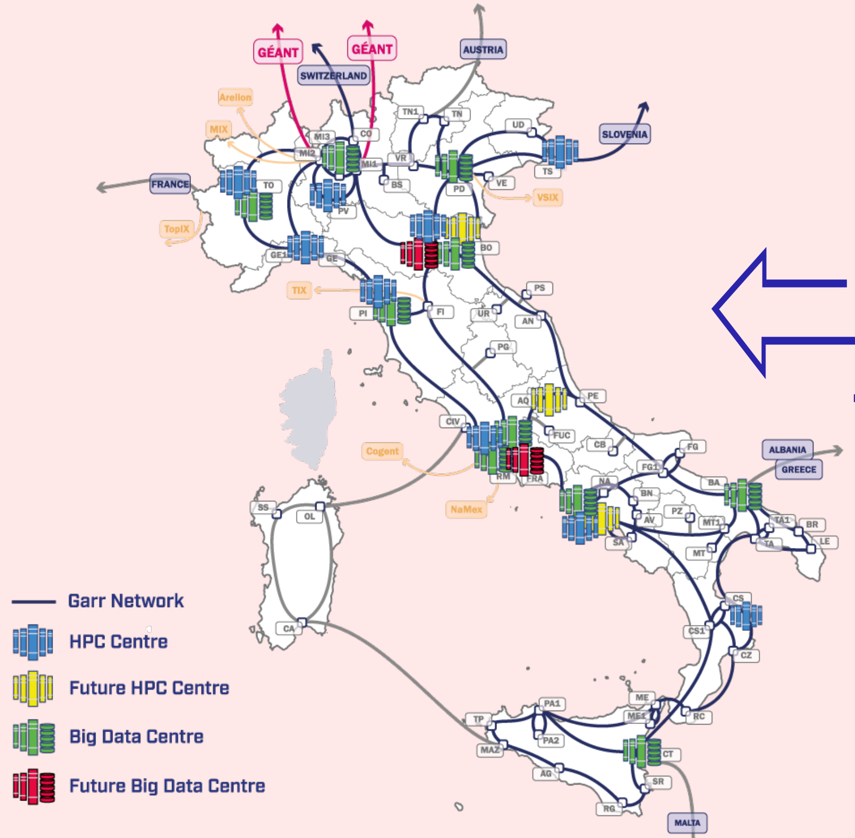
SOLUTION:
HPCs can fulfill the computing needs through the era of HL-LHC (Run 4) and DUNE.

See Charles Leggett's talk for more details.

U.S. DEPARTMENT OF ENERGY | Office of Science
Argonne NATIONAL LABORATORY
Brookhaven National Laboratory
Fermilab
BERKELEY LAB Bringing Science Solutions to the World

L'ICSC includes
10 thematic spokes
1 infrastructure spoke

0 SUPERCOMPUTING CLOUD INFRASTRUCTURE



High-level teams of experts integrating
the Spokes working groups (mixed cross-sectional teams)

ISTRUZIONE E FORMAZIONE, IMPRENDITORIALITÀ, TRASFERIMENTO DI CONOSCENZE, POLICY, OUTREACH

1 FUTURE HPC & BIG DATA

2 FUNDAMENTAL RESEARCH & SPACE ECONOMY

3 ASTROPHYSICS & COSMOS OBSERVATIONS

4 EARTH & CLIMATE

5 ENVIRONMENT & NATURAL DISASTERS

6 MULTISCALE MODELING & ENGINEERING APPLICATIONS

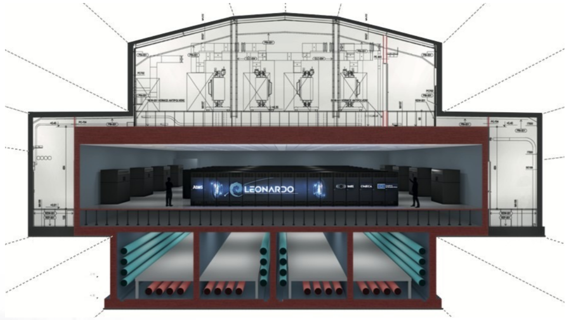
7 MATERIALS & MOLECULAR SCIENCES

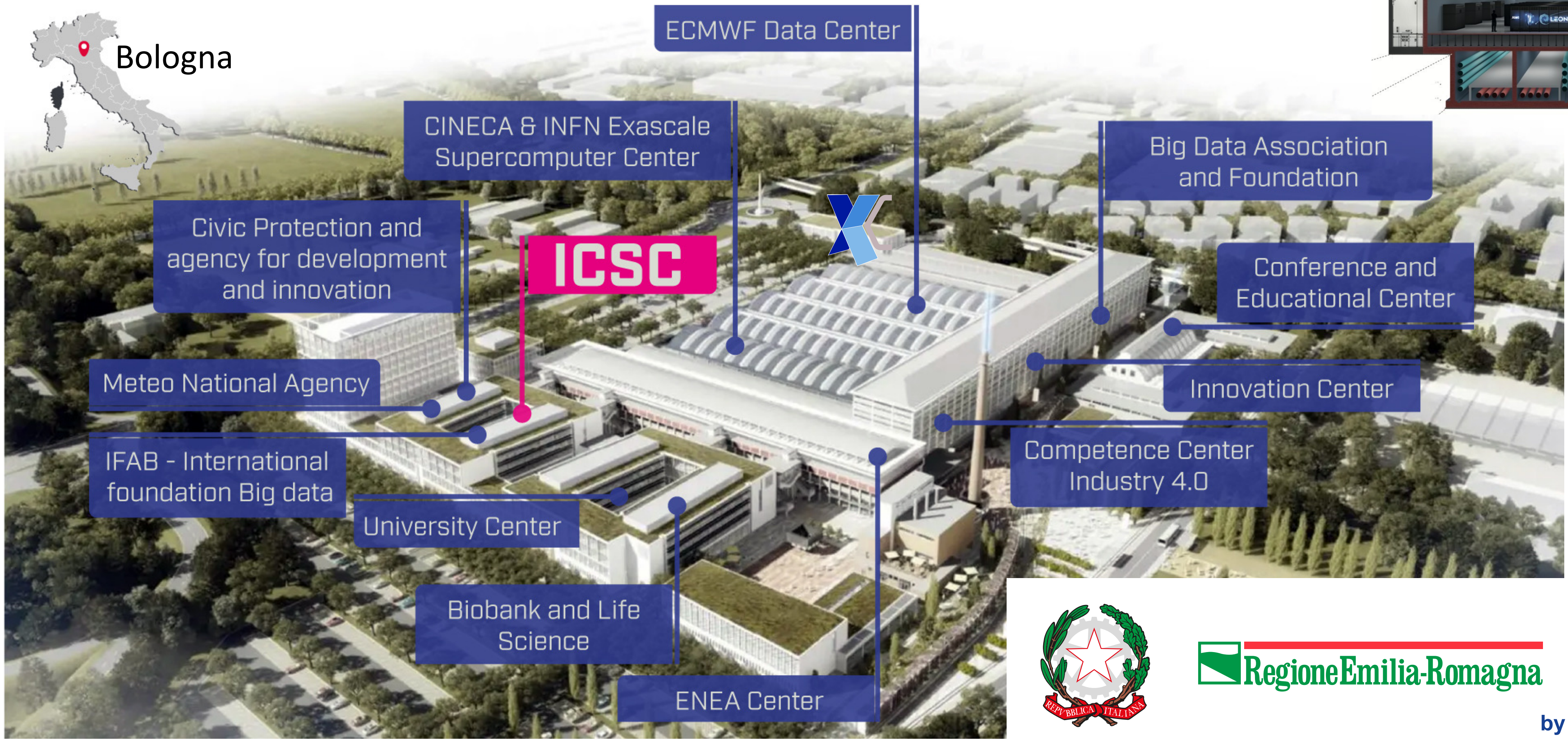
8 IN-SILICO MEDICINE & OMICS DATA

9 DIGITAL SOCIETY & SMART CITIES

10 QUANTUM COMPUTING

The Bologna Big Data Technopole

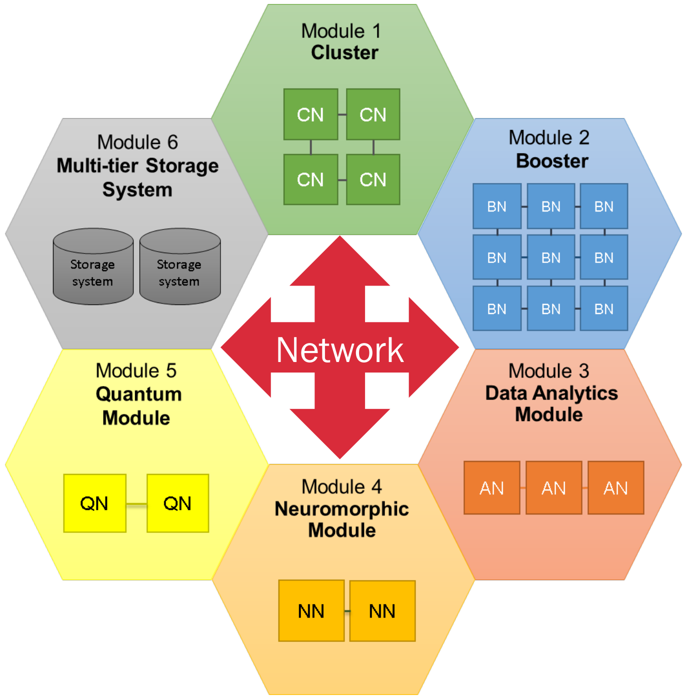


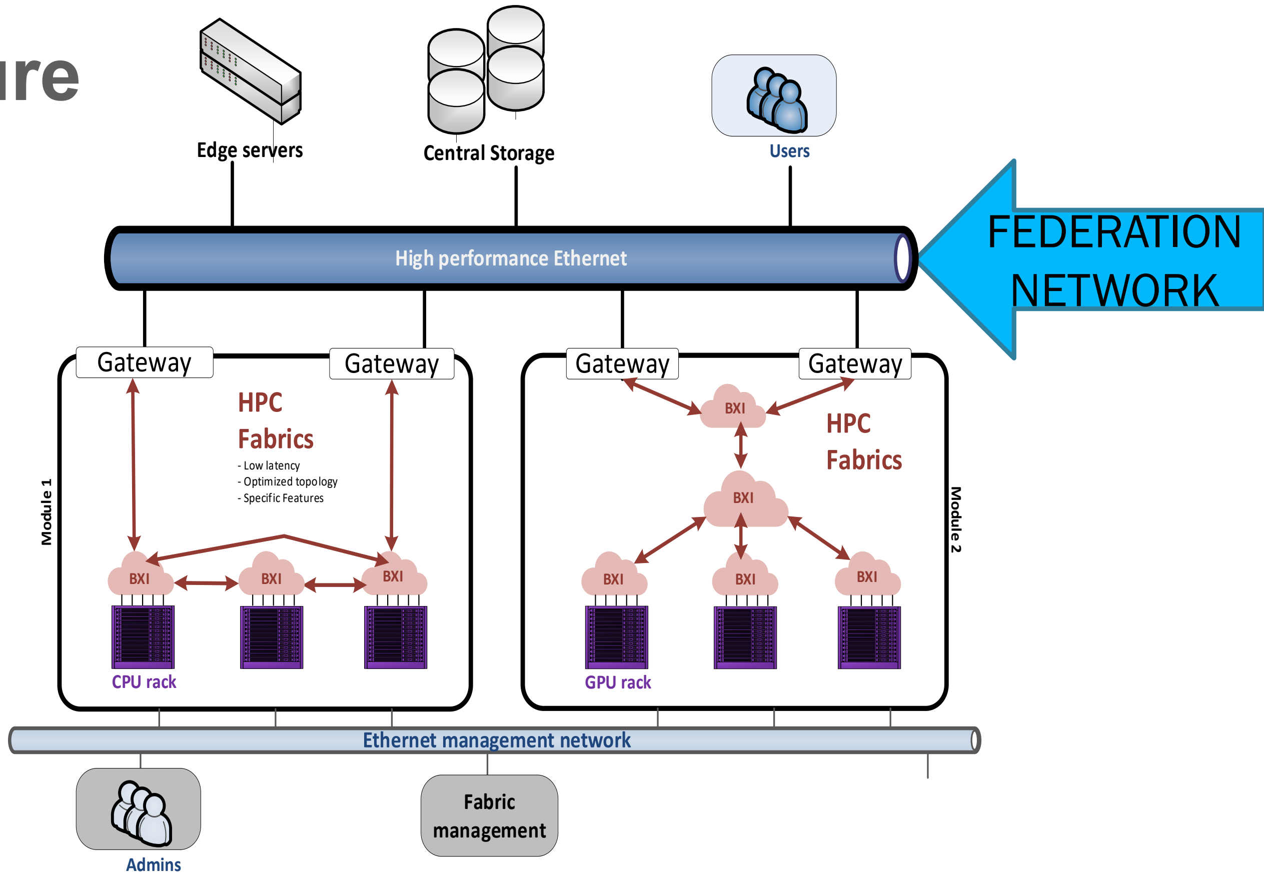


Regione Emilia-Romagna

Co-funded by the European Union

RED-SEA: MSA network architecture





Modular Supercomputer Architecture (MSA)

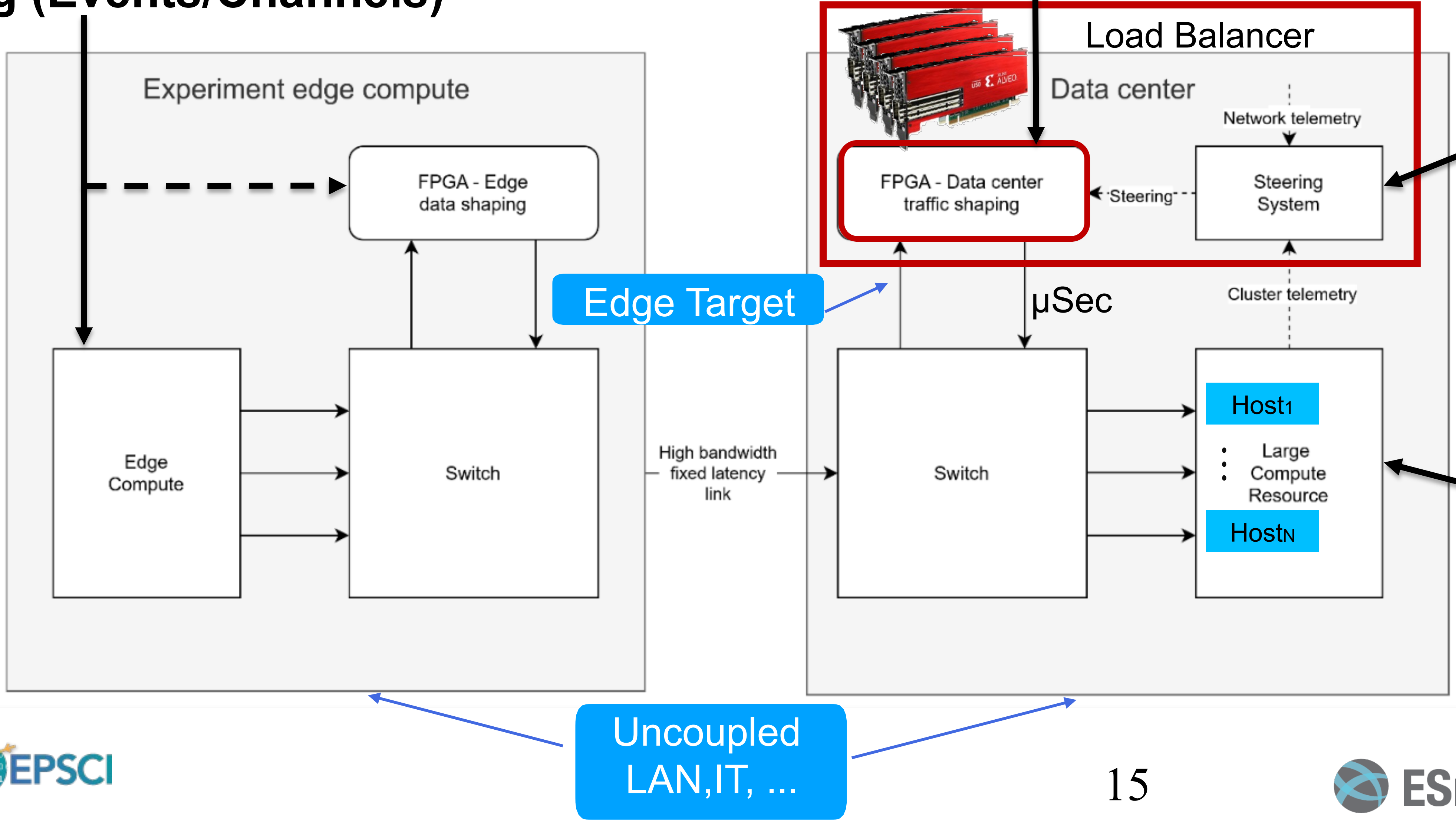
- aggregation of resources that are organized to facilitate the mapping of applicative workflows
  - HPC (High-Performance Computing)
  - HPDA (High-Performance Data Analytics)
  - AI (Artificial Intelligence)

- High performance Ethernet as federation network featuring state-of-the-art low latency RDMA communication semantics
- BXI as the HPC fabric consisting of two discrete components
  - a BXI NIC plus a BXI switch
  - the BXI fabric manager

Layer 3 Load Balancer

Compress, UDP Fragment, Tag (Events/Channels)

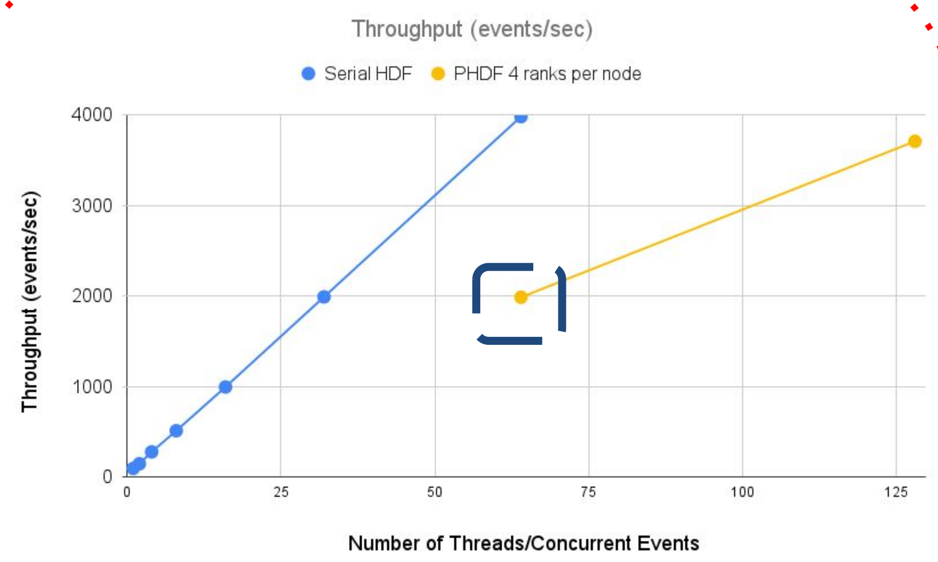
Rewrite UDP Packet Header



Dynamically Load Balance

Decompress, Reassemble, Process

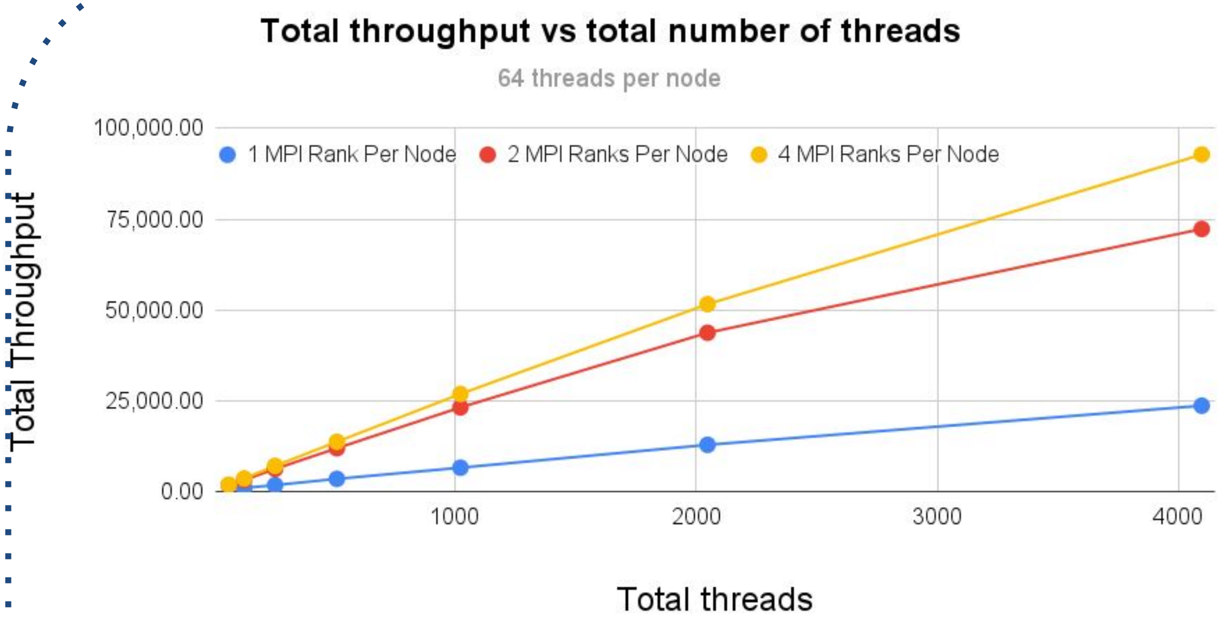
Parallel I/O with HDF5



Test done in a single node
Batch size of 100 events

Throughput = (Number of Events processed)/(Application Run time)

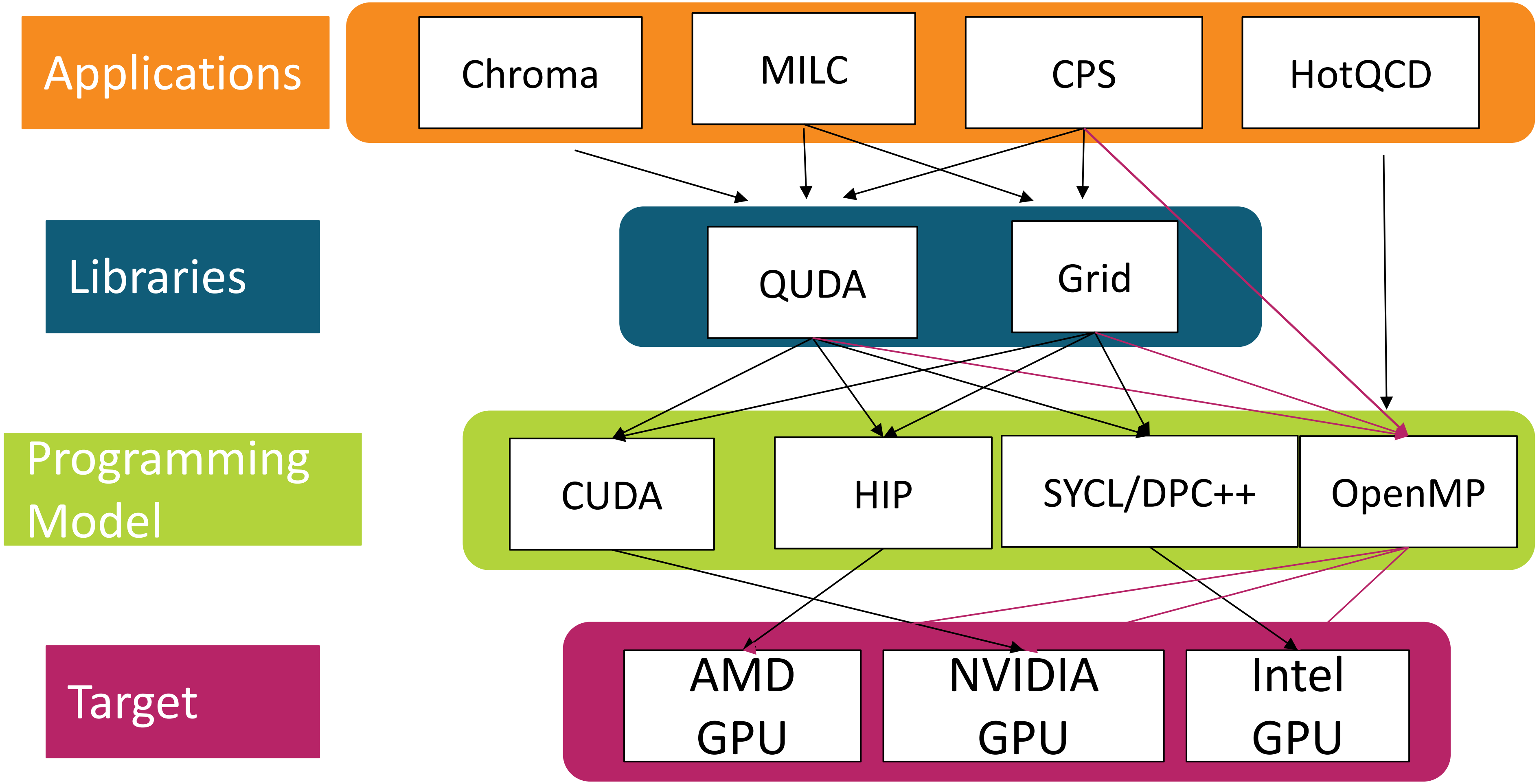
For Parallel I/O:
4 parallel processes
Threads per Rank: #Threads/4



- **Total Throughput**: (Throughput per rank) X(MPI Ranks)
- Test with **64 threads per node**.

I/O Calls	Fraction of Total I/O Time
MPI calls (external to HDF5)	14%
Write data into HDF5 file	32%
Other (including serialization)	54%

Exascale Lattice QCD Software Suite



Multi-pronged approach

Currently focused on architecture-specific programming models for best performance

Also exploring OpenMP offloading for better portability

Brookhaven National Laboratory

ECP Exascale Computing Project

Portable Parallelization Strategies (PPS)

	CUDA	Kokkos	SYCL	HIP	OpenMP	alpaka	std::par
NVIDIA GPU			*intel/llvm compute-cpp*	*hipcc*	*nvc++ LLVM, Cray GCC, XL*		*nvc++*
AMD GPU			*openSYCL intel/llvm*	*hipcc*	*AOMP LLVM Cray*		
Intel GPU			*oneAPI intel/llvm*	CHIP-SPV: *early prototype*	*Intel OneAPI compiler*	*prototype*	*oneapi::dpl*
x86 CPU			*oneAPI intel/llvm computecpp*	*via HIP-CPU Runtime*	*nvc++ LLVM, CCE, GCC, XL*		
FPGA				*via Xilinx Runtime*	*prototype compilers (OpenArc, Intel, etc.)*	*protytype via SYCL*	

Portable Parallelization Strategies

Ported representative testbeds from ATLAS, CMS and DUNE to each portability layer.

	Kokkos	SYCL	OpenMP	Alpaka	std::par
Patatrack	Done	Done*	WIP	Done*	Done compiler bugs
Wirecell	Done	Done	Done	no	Done
FastCaloSim	Done	Done	Done	Done	Done
P2R	done	Done	OpenACC	Done	Done

Evaluated each porting experience according to a number of different objective and subjective metrics.

Related talks:

- p2r [Monday 11AM]
- Patatrack [Thursday 12PM]
- FastCaloSim: overview [Tuesday 3PM]
- FastCaloSim: OpenMP (poster)
- FastCaloSim: alpaka + std::par (poster)

NVIDIA® OptiX™ Ray Tracing Engine -- Accessible GPU Ray Tracing

OptiX makes GPU ray tracing accessible

- **Programmable GPU-accelerated Ray-Tracing Pipeline**
- Single-ray shader programming model using CUDA
- ray tracing acceleration using RT Cores (RTX GPUs)
- "...free to use within any application..."

OptiX features

- acceleration structure creation + traversal (eg BVH)
- instanced sharing of geometry + acceleration structures
- compiler optimized for GPU ray tracing

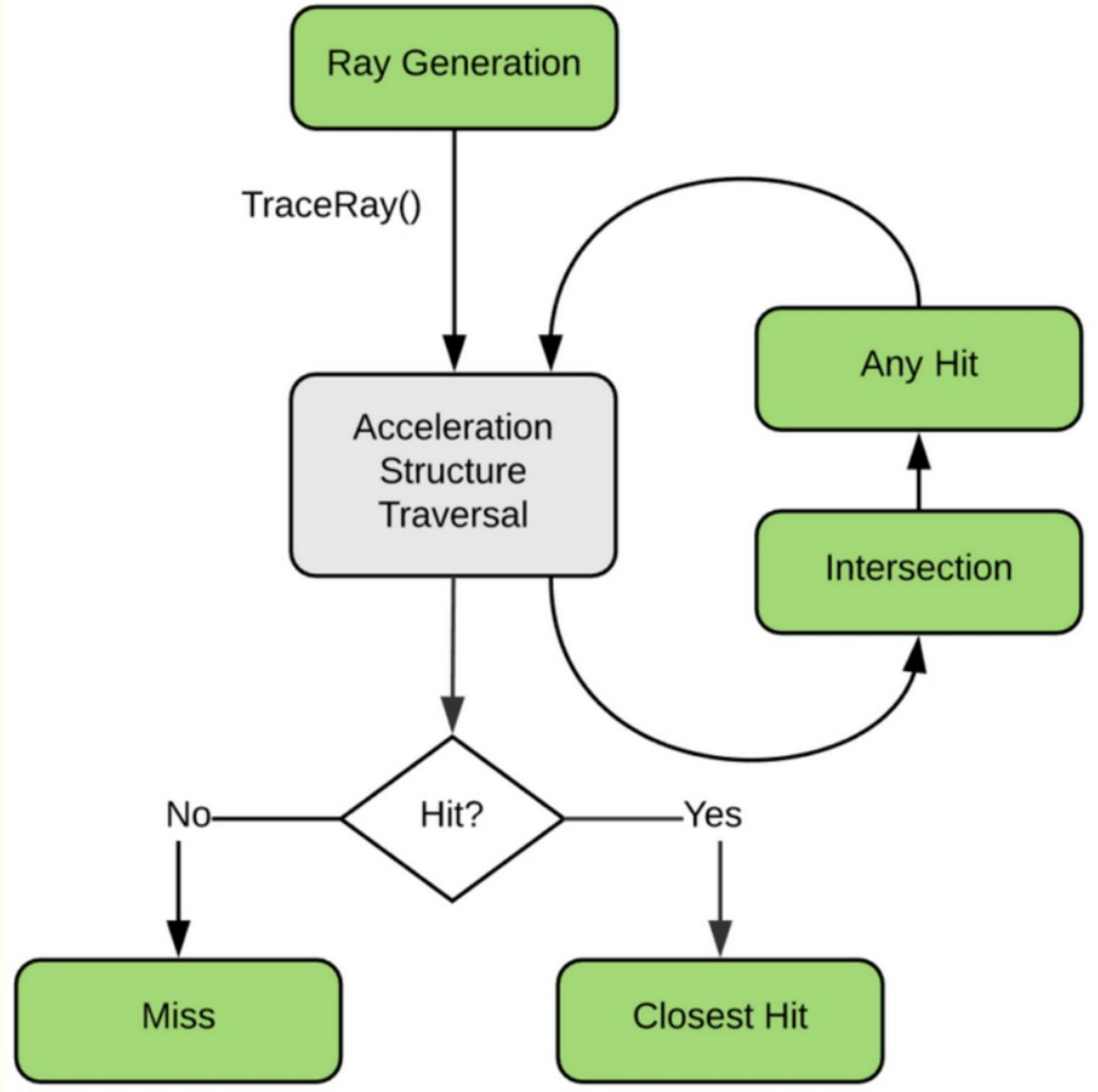
`https://developer.nvidia.com/rtx/ray-tracing/optix`

User provides (Green):

- ray generation
- geometry bounding boxes
- intersect functions
- instance transforms

Flexible Ray Tracing Pipeline

Green: User Programs, **Grey: Fixed function/HW**



Analogous to OpenGL rasterization pipeline

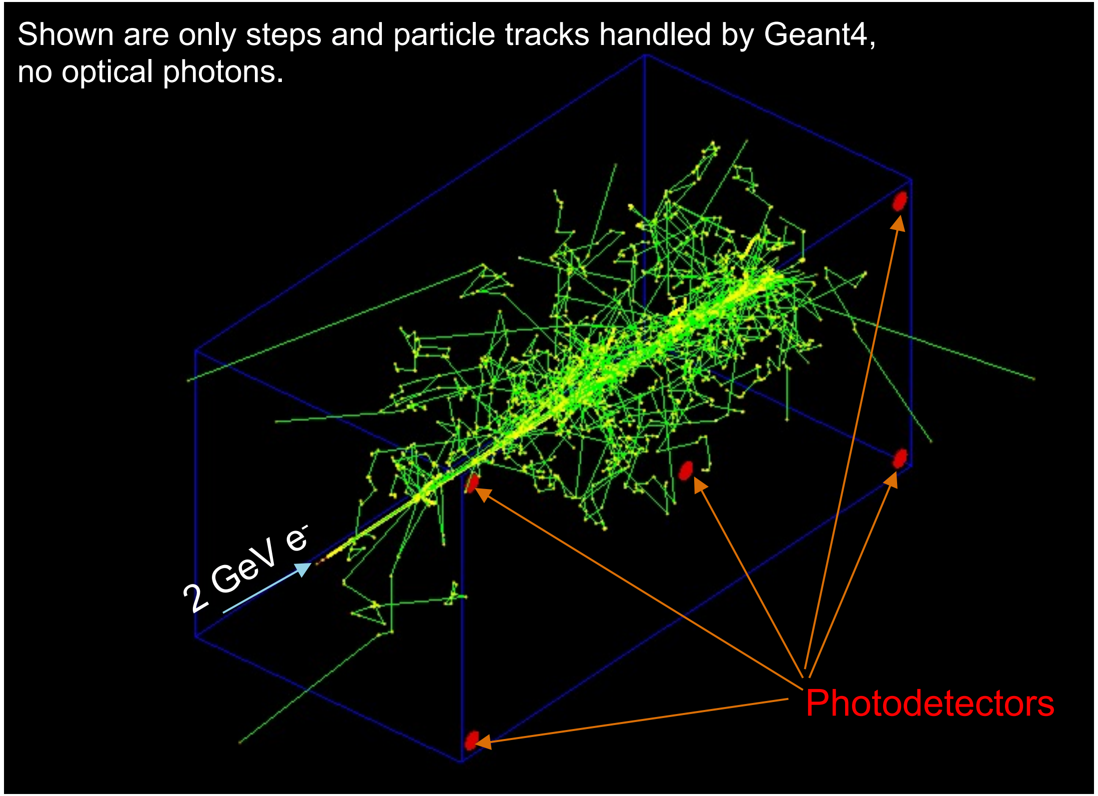
The computational challenge for TPCs based on liquid Argon (LArTPCs):

Test Detector Geometry:
Liquid Argon: x y z: 1 x 1 x 2 m (blue)
5 photo detectors (red)
photon yield (no E-field): 50000 γ/MeV
single 2 GeV electron (shower not fully contained)
(low Z=18, low $\rho = 1.78$ g/cm^3).

- **~ 7x10^7** VUV scintillation photons are produced/event.
- Using Geant4 (11.1.p01) to simulate photon generation and propagation o using a single core on an Intel® Core i9-10900k@ 3.7Ghz takes :

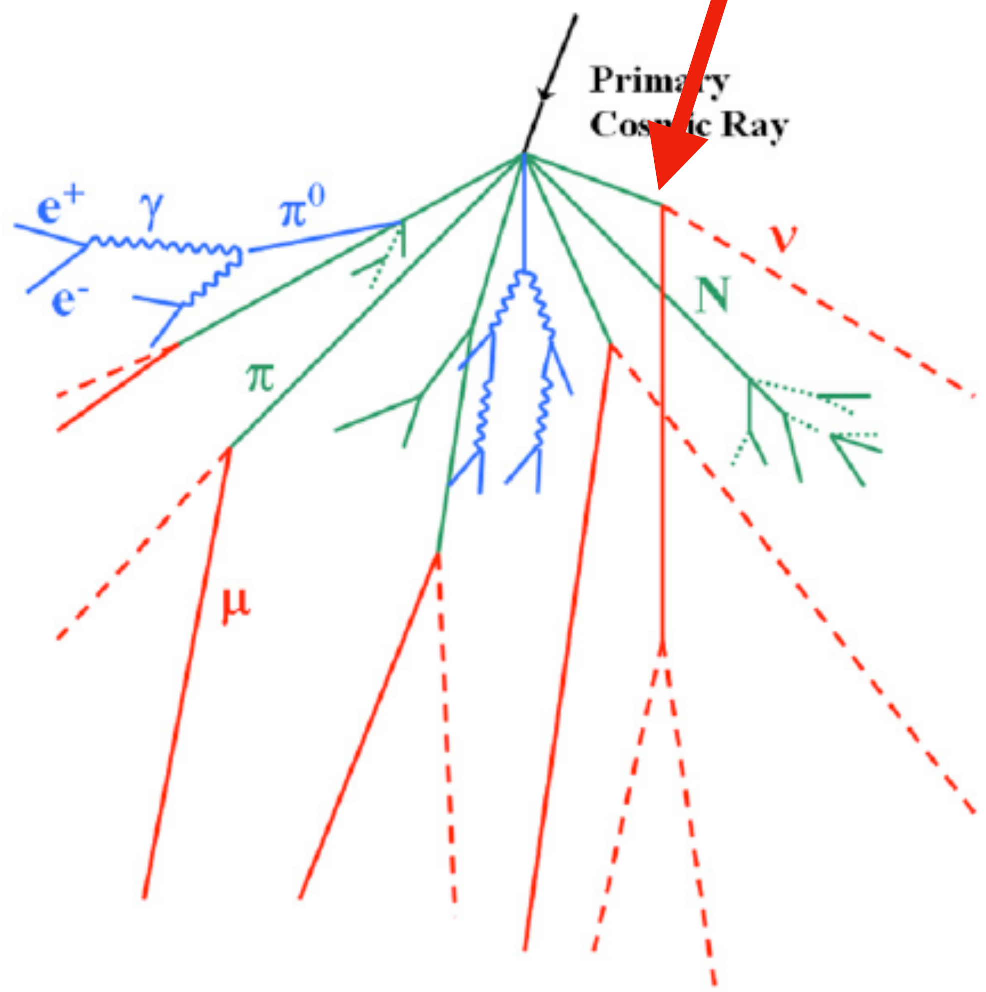
~ 10 minutes/event

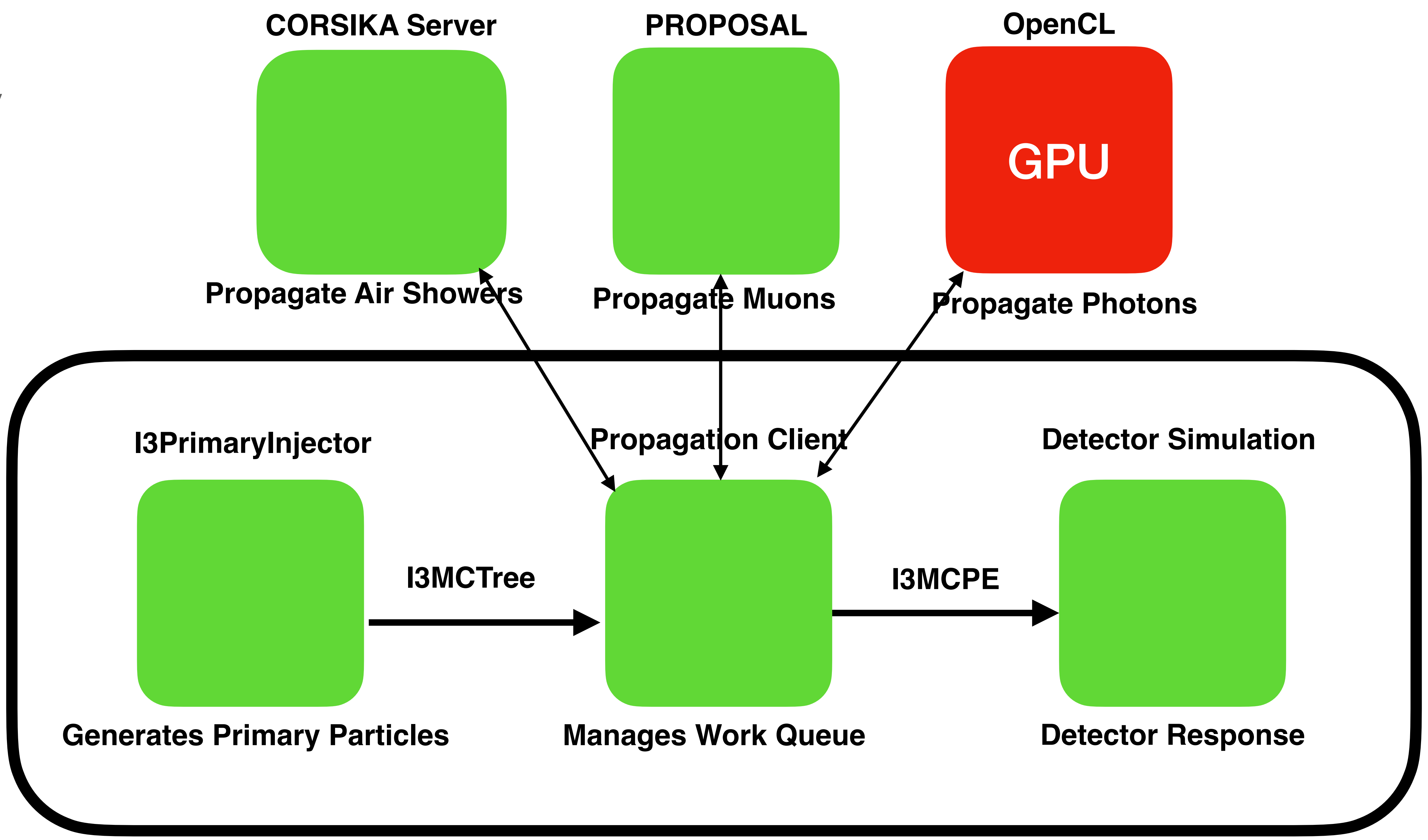
(Compared to **0.034 seconds/event** without optical photon simulation) → LArTPC-Experiments use look up tables and parameterizations instead of full simulation for photon response.

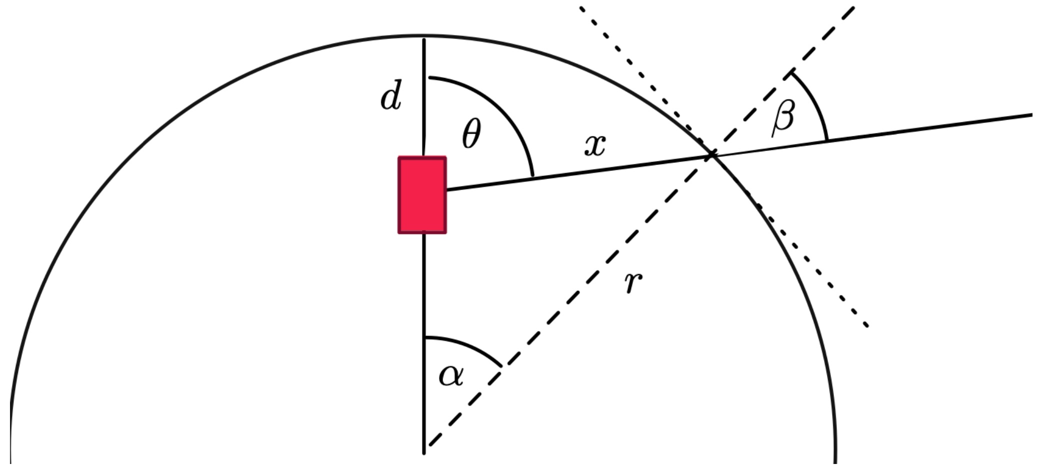


New Simulation Chain

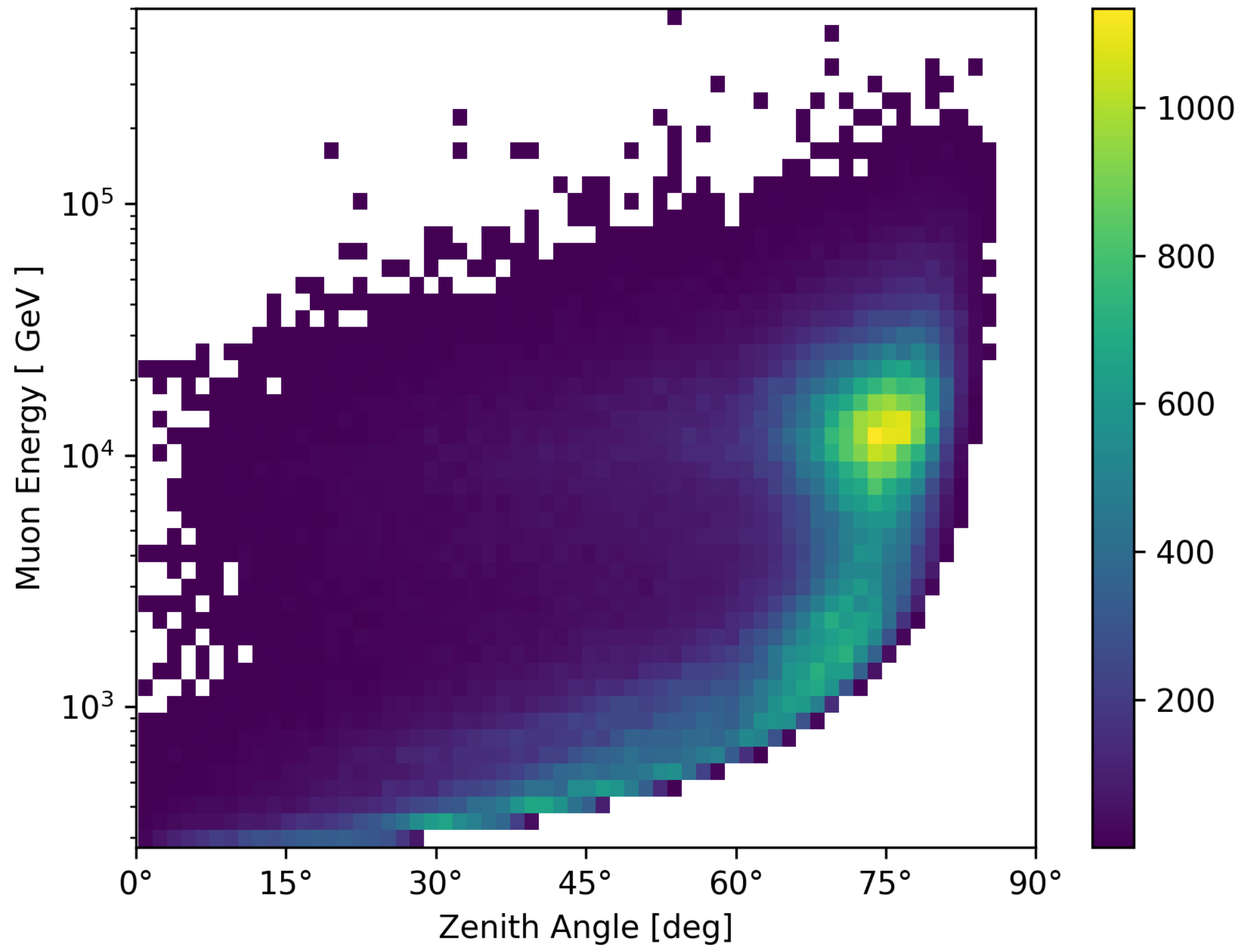
Leading Edge Muon is most likely to propagate all the way to the detector







More inclined showers encounter more matter between the surface and the detector and require higher energy muons to reach the detector



New CORSIKA Server will set the minimum muon energy higher for more inclined showers

CORSIKA Server

- CORSIKA was modified to run as a server which can be configured with individual primaries (thanks to D. Baack)
- Different CORSIKA cards for different showers
  - The energy needed to reach the detector is highly dependent on the inclination of the shower
  - The energy at which CORSIKA will stop propagating particles is now higher for inclined showers
- Showers with higher energy leading edge muons are undersampled:
  - Showers with low energy muons are killed before the rest of the shower is calculated
- Since individual particles are sent over IPC no files are written avoiding IO bottleneck

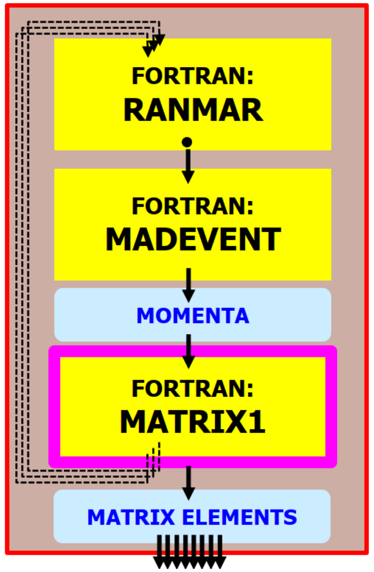
Celeritas version `0.3-dev`: Geant4 integration status

- **Imports** EM physics selection, cross sections, parameters
- **Converts** geometry to VecGeom model
- **Offloads** EM tracks from Geant4
- **Scores** hits to user “sensitive detectors”
- **Includes** GPU-optimized simple calorimeter
- **Integrates** with Geant4 10.6–11.0
- **Supports** physics/geometry/setup changes at link/run time

Celeritas is not designed to be a prototype code

MG5aMC: old and new architecture designs

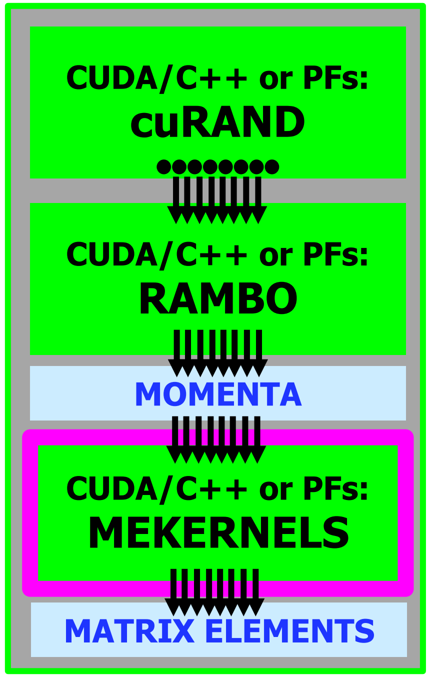
OLD MADEVENT
(CURRENT: LHC PROD)
SINGLE-EVENT API



MATRIX ELEMENT: CPU BOTTLENECK IN OLD MADEVENT

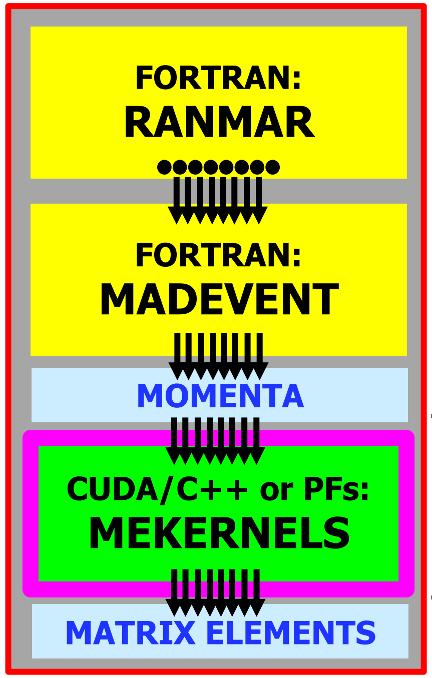
First we developed the new ME engines in standalone applications

1. STANDALONE
(TOY APPLICATIONS)
MULTI-EVENT API



Then we modified the existing all-Fortran MadEvent into a *multi-event* framework and we injected the new MEs into it

2. NEW MADEVENT
(GOAL: LHC PROD)
MULTI-EVENT API



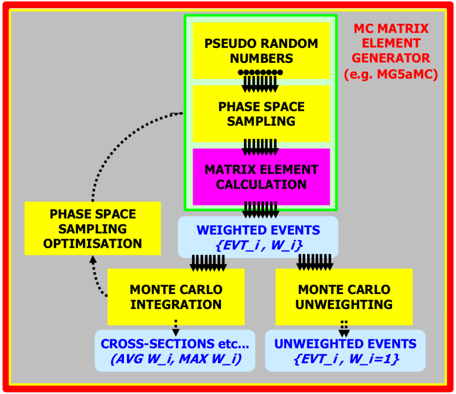
(Amdahl...)

SCALAR: NEW BOTTLENECK?

PARALLEL: MUCH FASTER!

Reweighing

- If new physics doesn't affect later simulation stages:
  Only need to regenerate events
  *(Note: **Very** hard condition)*
- $\rightarrow$ Can recycle simulations
  (Note: Must simulate orig. evt.)
- Furthermore: Event generation factorises, too



Imperial College
London

Monte Carlo simulations

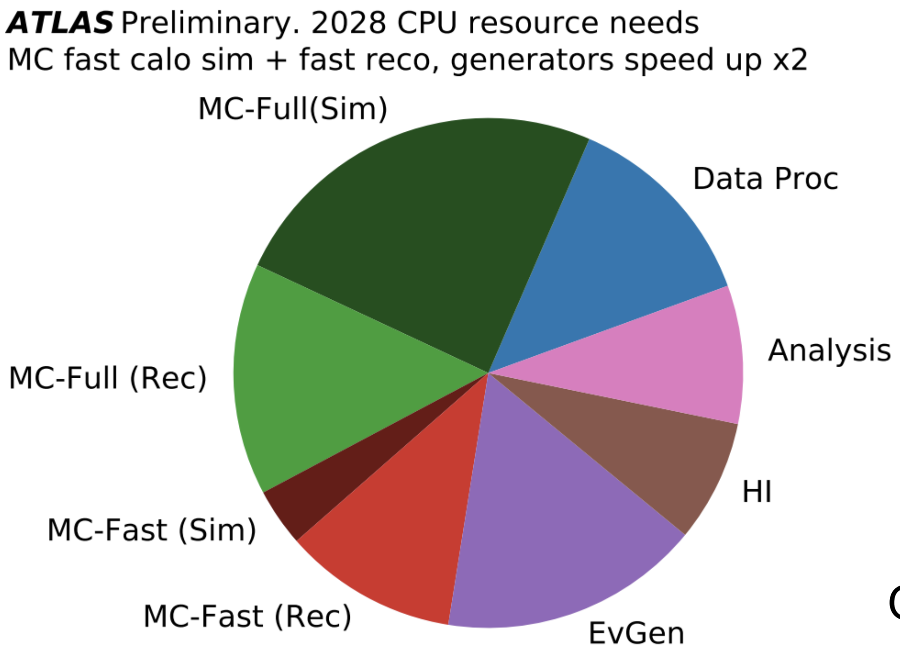


Figure 2: Breakdown of estimated compute workloads in 2028 for ATLAS

Over 50% is required by Monte Carlo related workloads

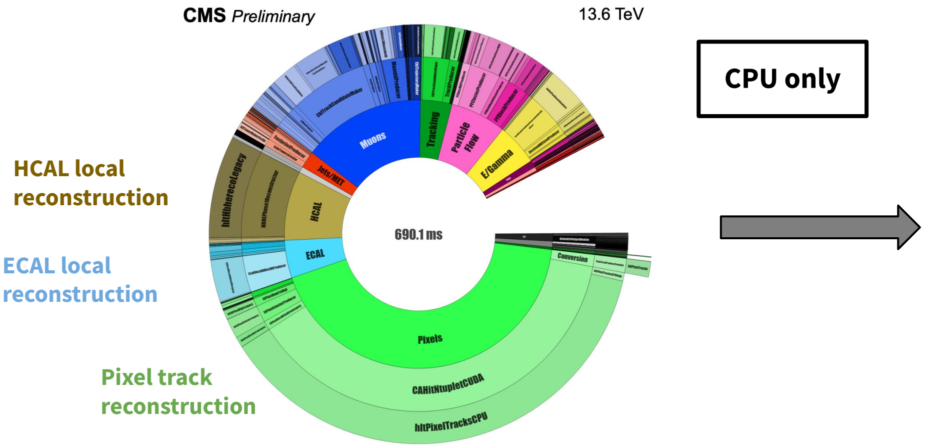
https://doi.org/10.22323/1.390.0009

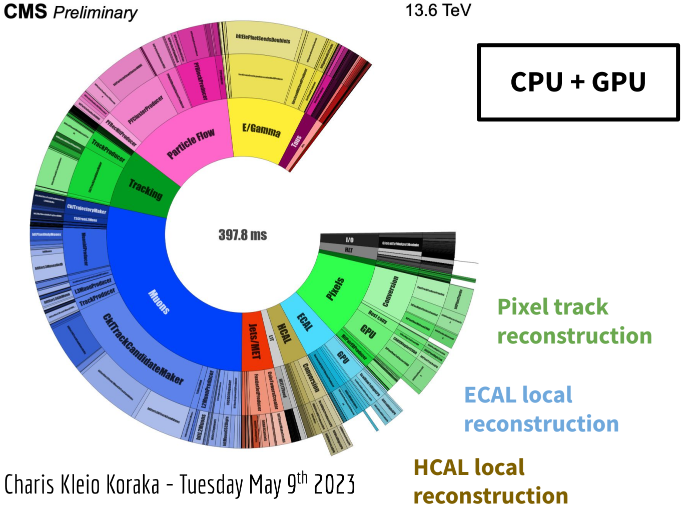
GPUs at the High Level Trigger

CMS has leveraged GPUs for the online reconstruction at High Level Trigger (HLT) starting from the beginning of Run-3 (2022-today)

What has been offloaded to GPU :

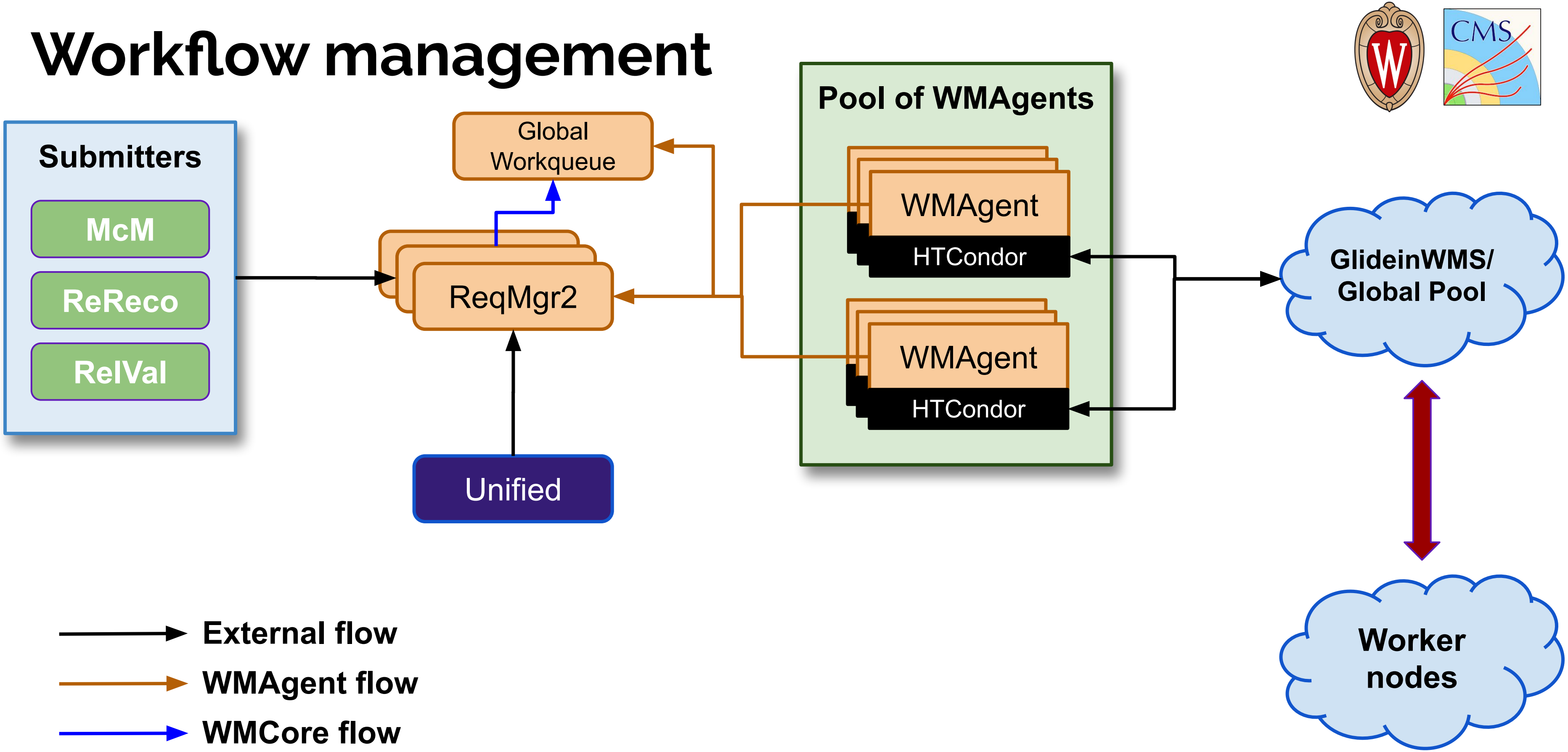
- ~25% of online reconstruction:
  - **Pixel track** reconstruction **ECAL** & **HCAL** local reconstruction





*Source CERN-CMS-DP-2023-004

Workflow management



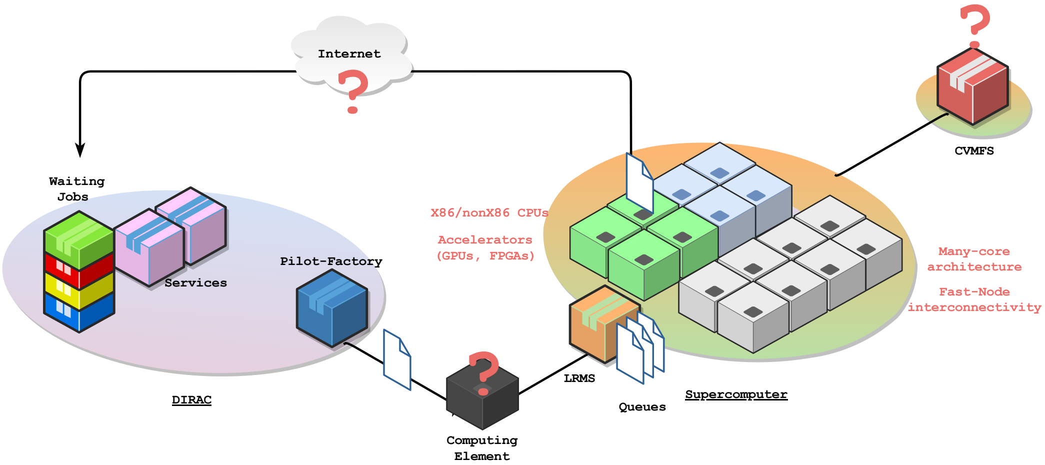
So what 'stuff' can we throw away?

- The problem is no longer one of rejecting (trivial) background
- Fundamentally changes what it means to trigger

- Instead, we need to categorise different 'signals'
  - Requires access to as much of the event as possible, as early as possible
  - Solution: Drop the L0 trigger, reconstruct 30 MHz of events before making trigger decisions!

Introduction
The LHCb detector
Upgrade 1
Why GPUs
DAQ
GPU Performance
Upgrade 2
Conclusions

DIRAC Workload Management System & Supercomputers?



Fast Calorimeter Simulation for GPU Portability Studies

ATLAS needs lots of simulation

- Simulation for background modeling is paramount for precision physics
- Lack of MC-based statistics limited results in Run-2
  - will be worse for Run-3 and beyond

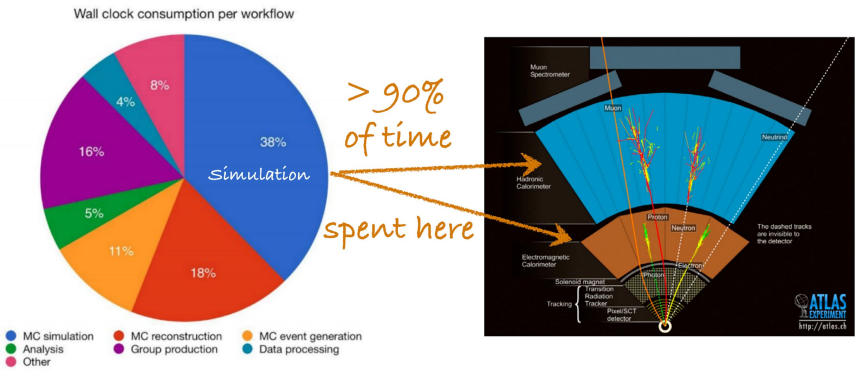
A very large fraction of the simulation's computational budget is spent in the LAr Calorimeter

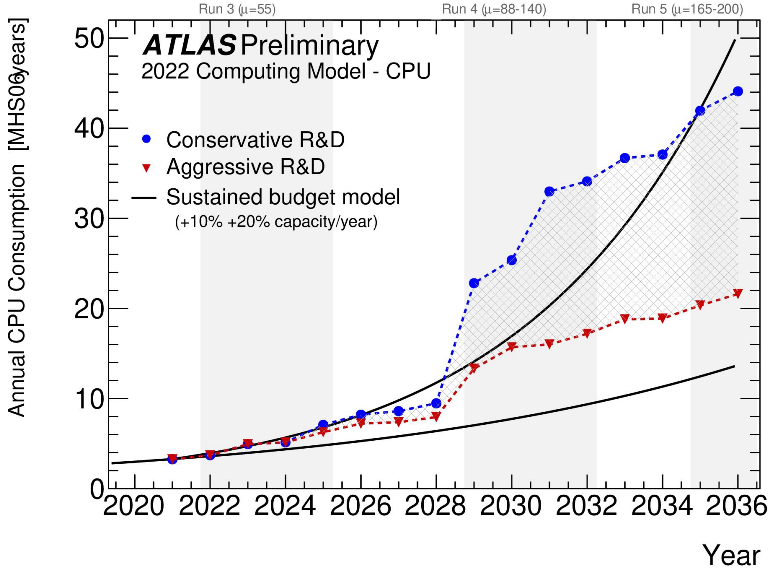
- Parametrized simulation is enormously faster than full Geant4 simulation (complex detector geometry)

FastCaloSim is small, self-contained, has few dependencies, and already has a CUDA port

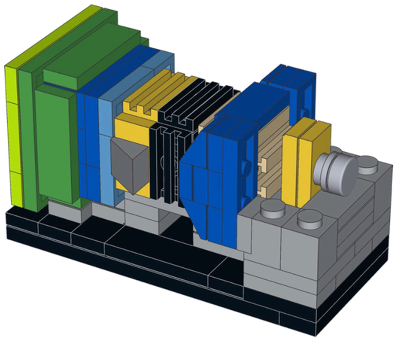
- Offloading simulation to GPUs can help stay within ATLAS's compute budget
- 3 "kernels": workspace reset, simulate, reduce plus small data transfers from device to host
- Code organized to share maximum functionality between all implementations

Calorimeter-dominated



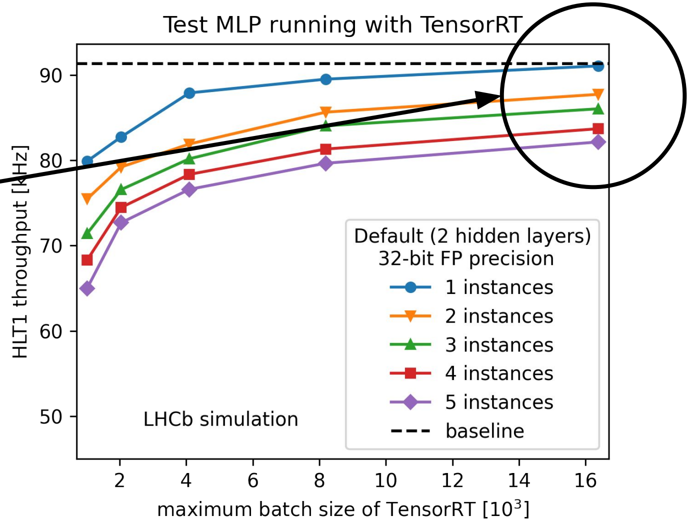


Conclusions and outlook

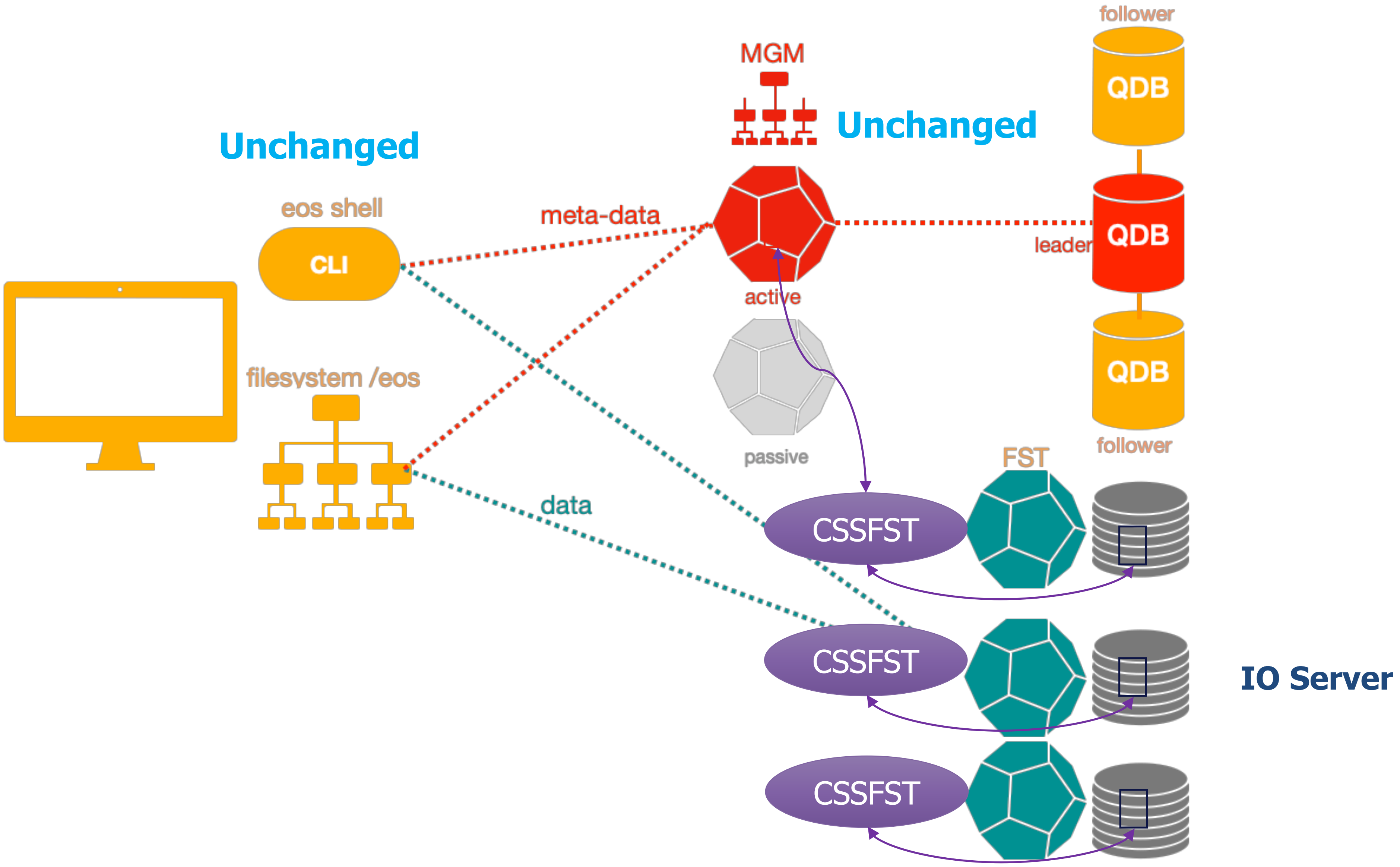


- **LHCb** has **high demands of throughput** of reconstruction and selection on GPUs to cope with high signal rates
- **Machine learning ideal to reduce rates while keeping signal efficiencies high**

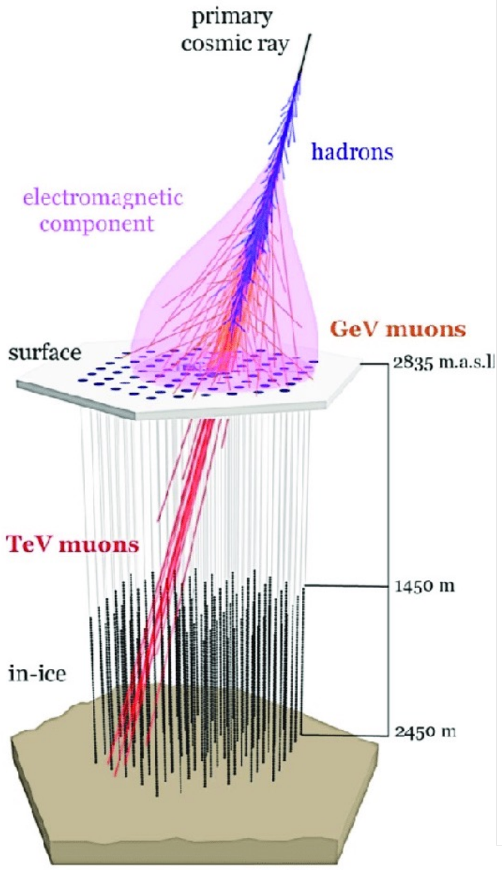
- Introducing **flexible loading of ML models** at the **first trigger level** (running on GPUs) with **TensorRT**
  - Multiple copies of typical sized MLPs seems to **effect throughput in an acceptable way**
- **Promising avenue of having flexible ML reconstruction and selection at the first trigger level!**

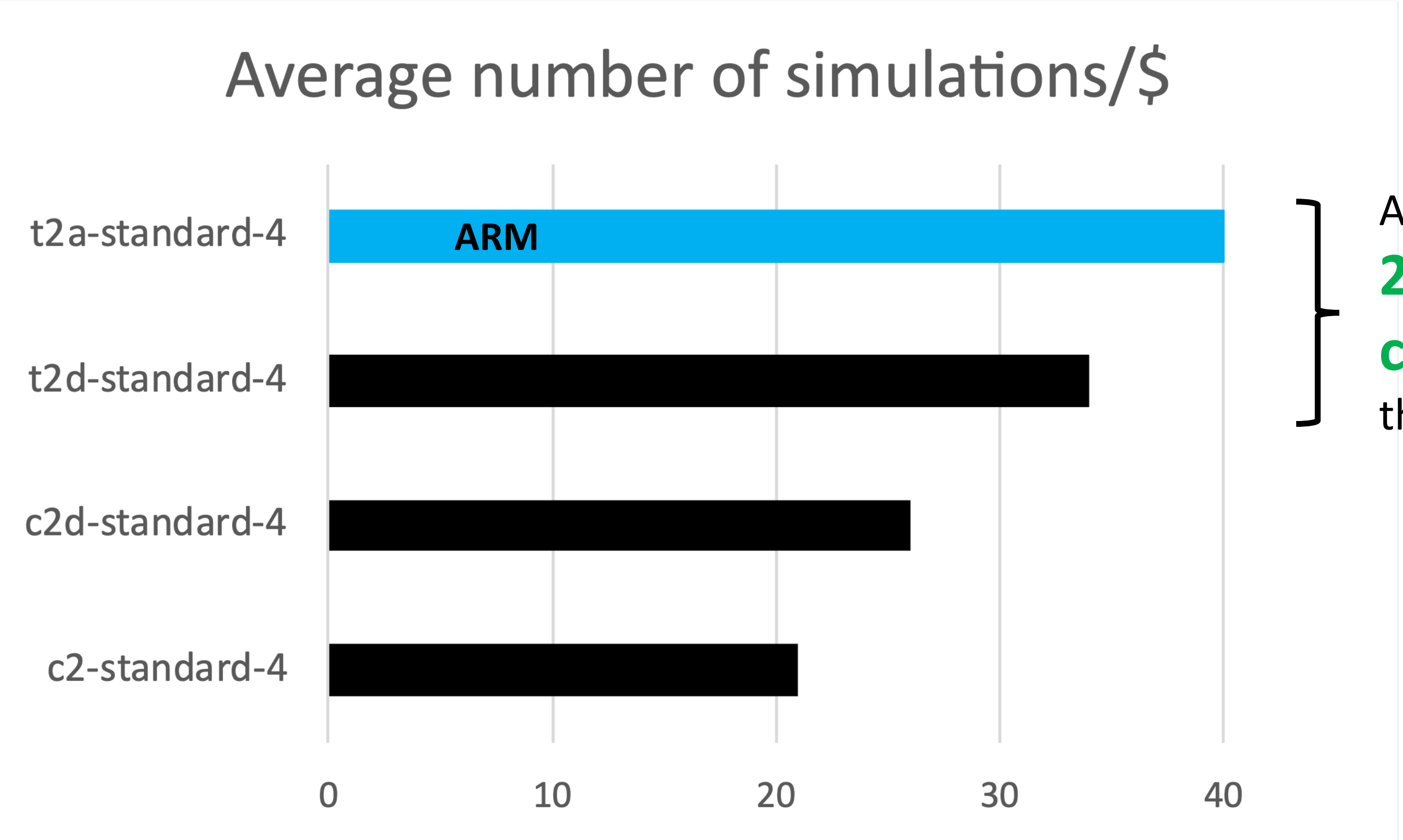


Architecture



Cost effectiveness





ARM about
20% more cost effective
that the closest x86

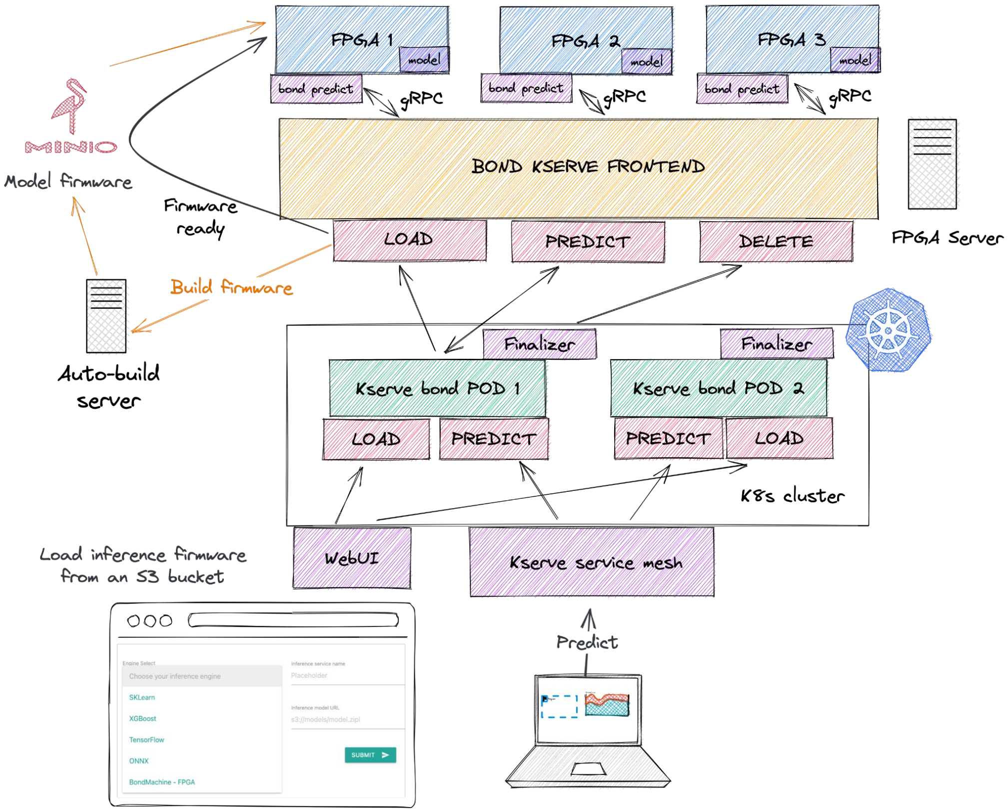
ARM CPUs more cost-effective than x86 CPUs (at least in early 2022)

Kserve extension implementation

We tested workflows for both small and ML board (ebaz4205, zedboard, alveo u50)

The main components that we developed are:

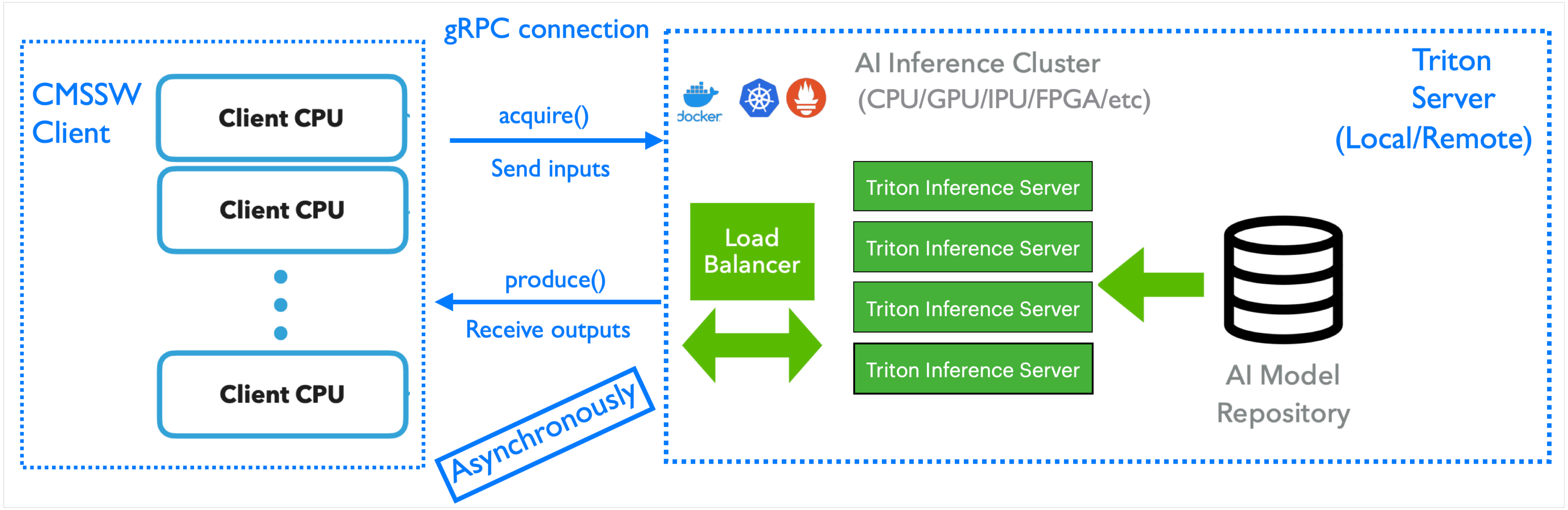
- **Custom WebUI** to hide complexity to the user
    - A Kubeflow managed solution exists, we are planning to integrate this work eventually
        - We need additional metadata to be passed (e.g. board model, provider, hls engine etc)
- Translate a **model load** request into conditional actions
    - Load the bitstream file from the remote location directly
        - Pre built by the user on its own
    - **building a firmware** “seamlessly” on an external building machine
- **Eventually load the firmware** on the FPGA board via the development of a grpc server installed on the machine that have access to the board



D.Ciangottini - ciangottini@infn.it - CHEP2023

SONIC

- Within CMS software (CMSSW), the IaaS deployment scheme is called "Services for Optimized Network Inference on Coprocessors" (SONIC)



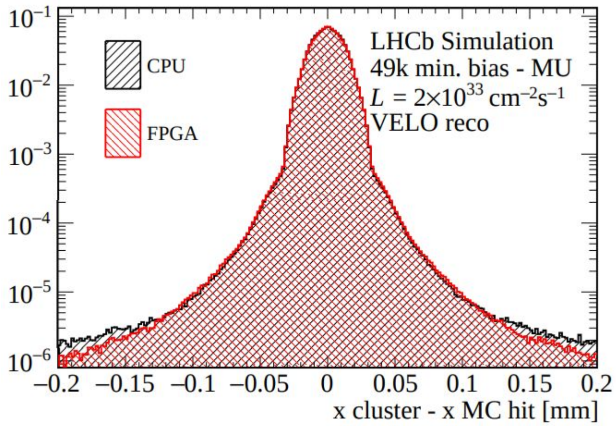
10.1109/TNS.2023.3273600

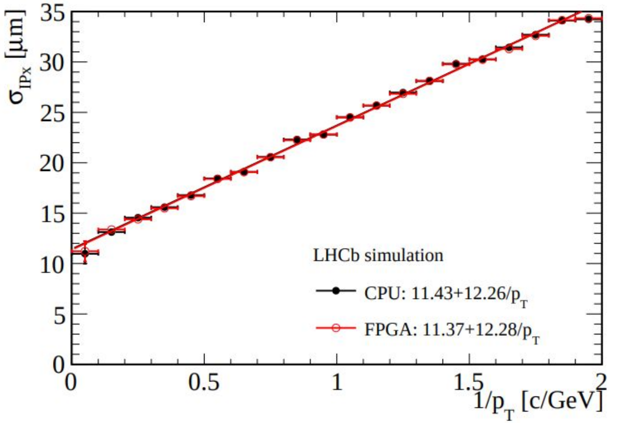
Physics performance

- Moving from a full-fledged software implementation of the VELO clustering to a FPGA-based one required a careful evaluation of possible impacts on physics performances in terms of
  - Cluster efficiency → find hit on detector
  - Cluster residual → match hit position
  - Track efficiency → find track
  - Track resolution → match track parameters

Track type	Quantity	CPU clusters [%]	FPGA clusters [%]
All VELO tracks	efficiency	98.254 ± 0.007	98.254 ± 0.007
	clone	1.231 ± 0.006	1.234 ± 0.006
Long tracks	efficiency	99.252 ± 0.006	99.252 ± 0.006
	clone	0.806 ± 0.006	0.806 ± 0.006
	ghost	0.848 ± 0.003	0.928 ± 0.003

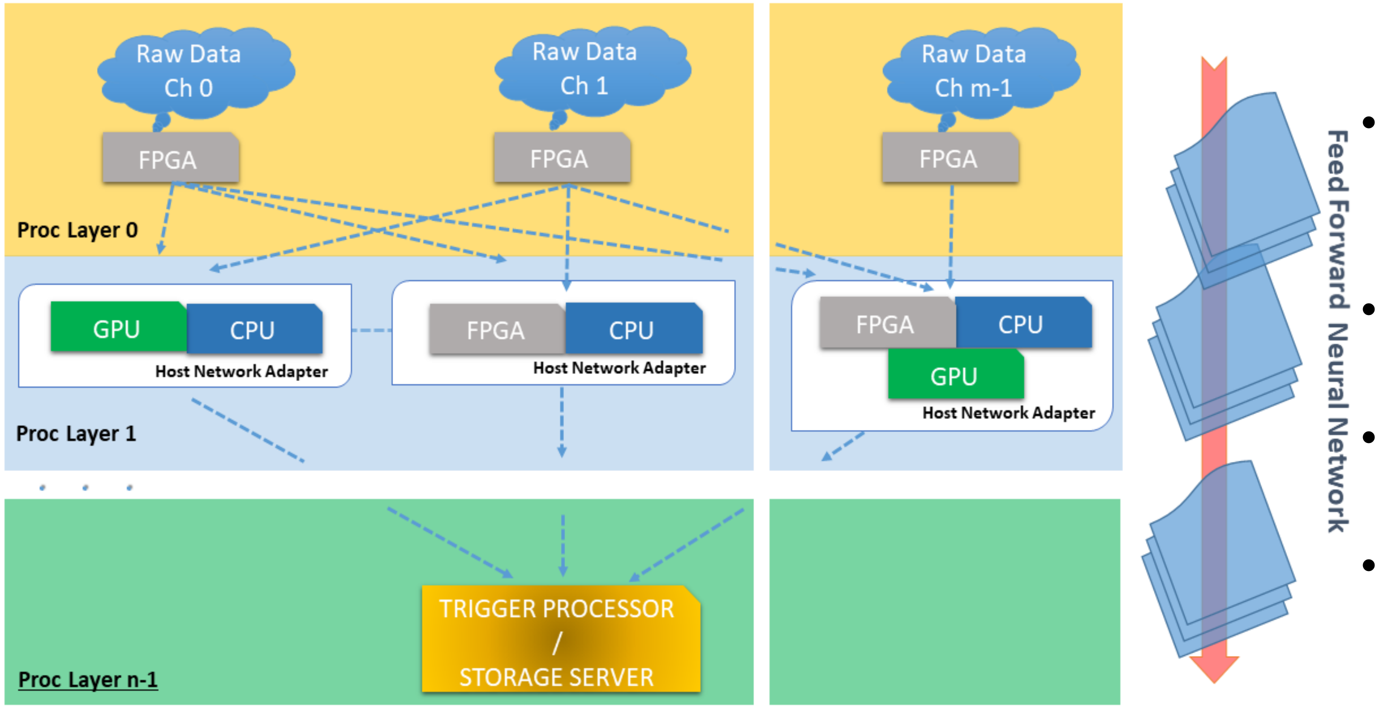
- **FPGA algorithm tracking performance is nearly indistinguishable from CPU/GPU clustering**





APEIRON in Trigger and Data Acquistion Systems

Abstract **P**rocessing **E**nvironment for **I**ntelligent **R**ead-**O**ut systems based on **N**eural networks



- Input data from several different channels (data sources, detectors/sub-detectors).
- **Data streams** from different channels recombined through the processing layers using **a low-latency, modular and scalable network infrastructure**
- Distributed online processing on heterogeneous computing devices (FPGAs for the moment) in *n* subsequent layers.
- Typically features extraction will occur in the first NN layers on RO FPGAs.
- More resource-demanding NN layers can be implemented in subsequent processing layers.
- Classification produced by the NN in last processing layer (e.g. pid) will be input for the **trigger processor/storage online data reduction stage for triggerless systems.**

Overview of Quantum Algorithms for NHEP

“Low-Level” Algorithms

- Grover’s & Shor’s algorithms
- Provable speedup / error correction required

Quantum Simulation

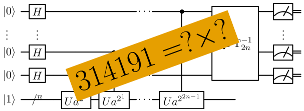
- Mimic system using simplified model
- Classically likely intractable

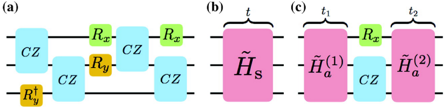
Unorthodox Approaches

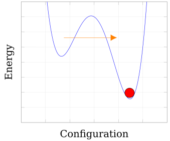
- Quantum annealing, adiabatic quantum computing
- (Gaussian) Boson sampling, etc.

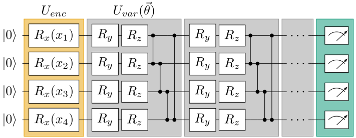
NISQ Algorithms

- Variational algorithms: Hybrid quantum-classical
- Less resources / potential speedups





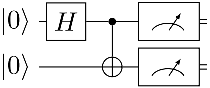




Overview of Quantum HW (high level)

Transmon	Ion Traps	Neutral Atoms
IBM, Google, rigetti	IONQ, AQT, QUANTINUUM, Honeywell	PASQAL, atom computing, ColdQuanta

$\Rightarrow$ Strong coupling to hardware properties

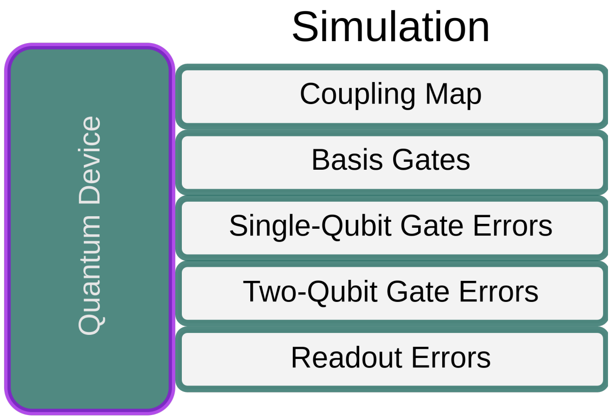


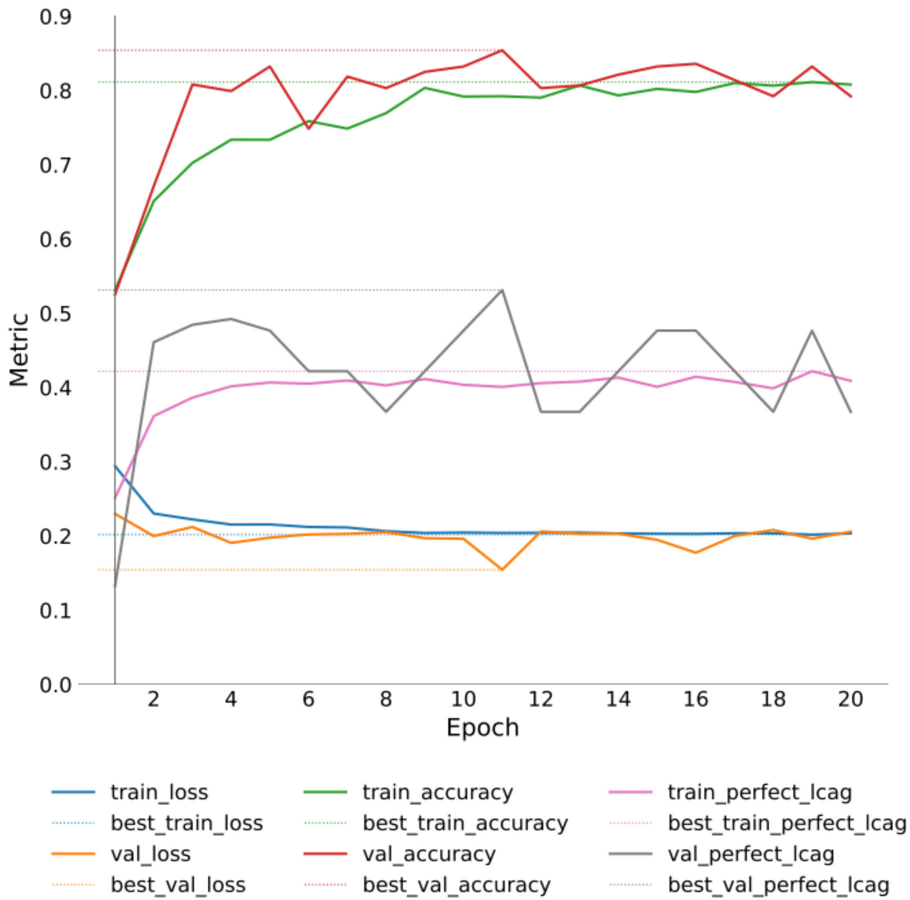
$\Leftrightarrow$

```
// prepare Bell state

OPENQASM 2.0;
include "qelib1.inc";

qreg q[2];
creg c[2];
reset q[0];
reset q[1];
h q[0];
cx q[0], q[1];
```

Results Noisy Simulation

- Noise-free Simulation:
  - Validation Accuracy: 0.857 (Step 11)
  - Validation Loss: 0.155 (Step 11)
- **Noisy Simulation**:
  - Device: IBM Perth
  - Validation Accuracy: 0.854 (Step 11)
  - Validation Loss: 0.154 (Step 11)

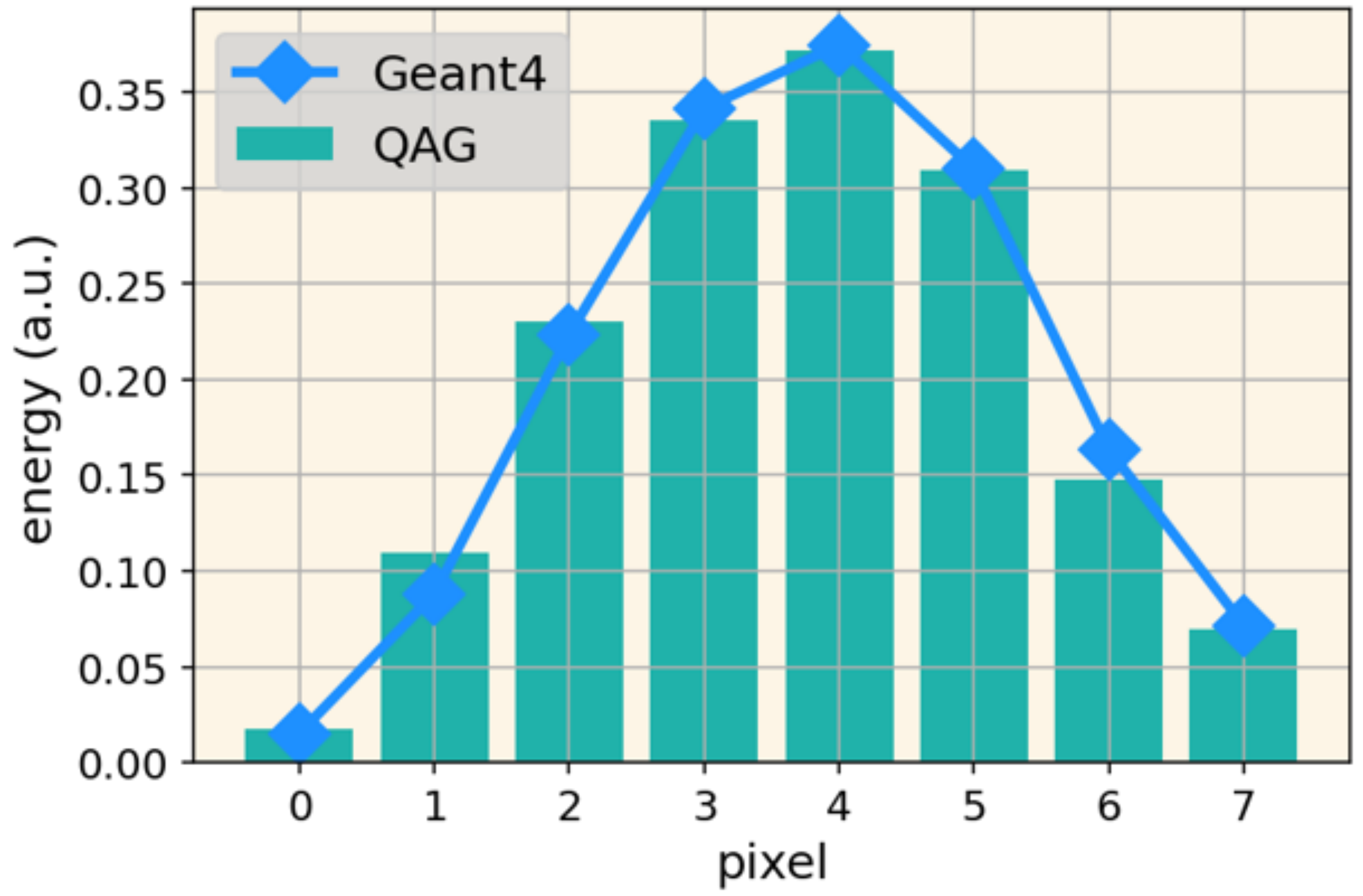




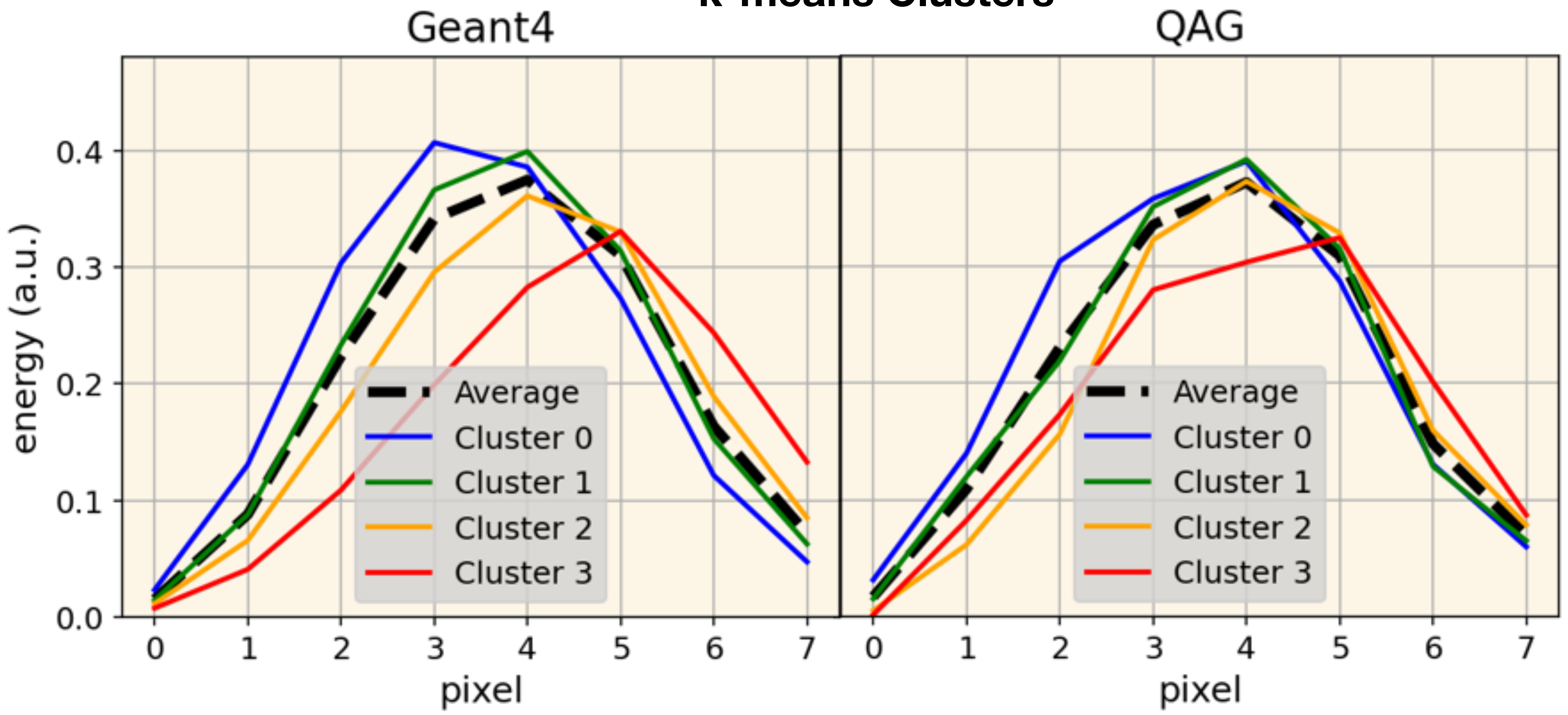
1. Use Case
2. QAG Model
3. Architecture
4. Training
5. Inference
6. Quantum Noise Study
6.1. Inference
6.2. Training
7. Conclusions

5. Inference

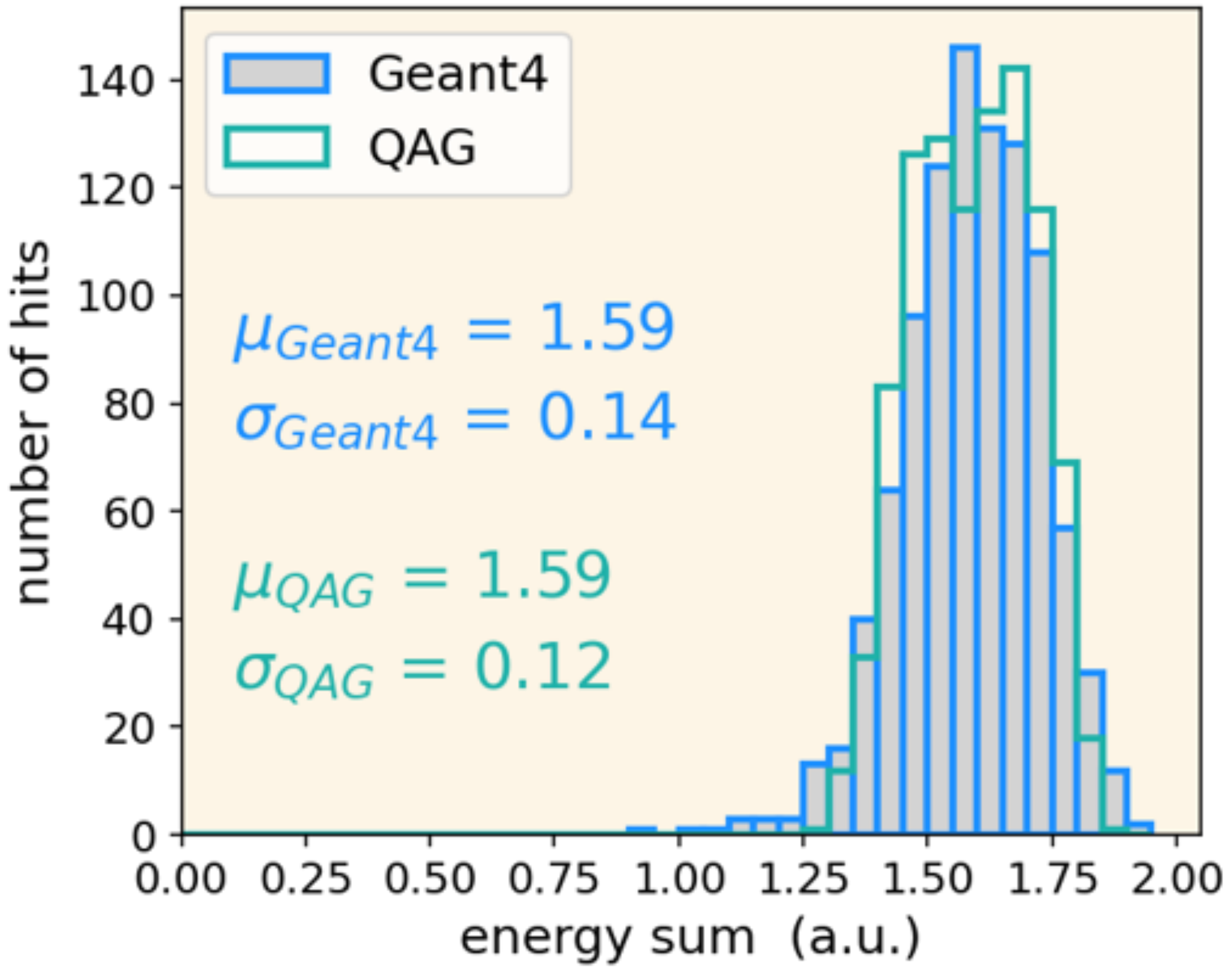
Average Calorimeter shower shape



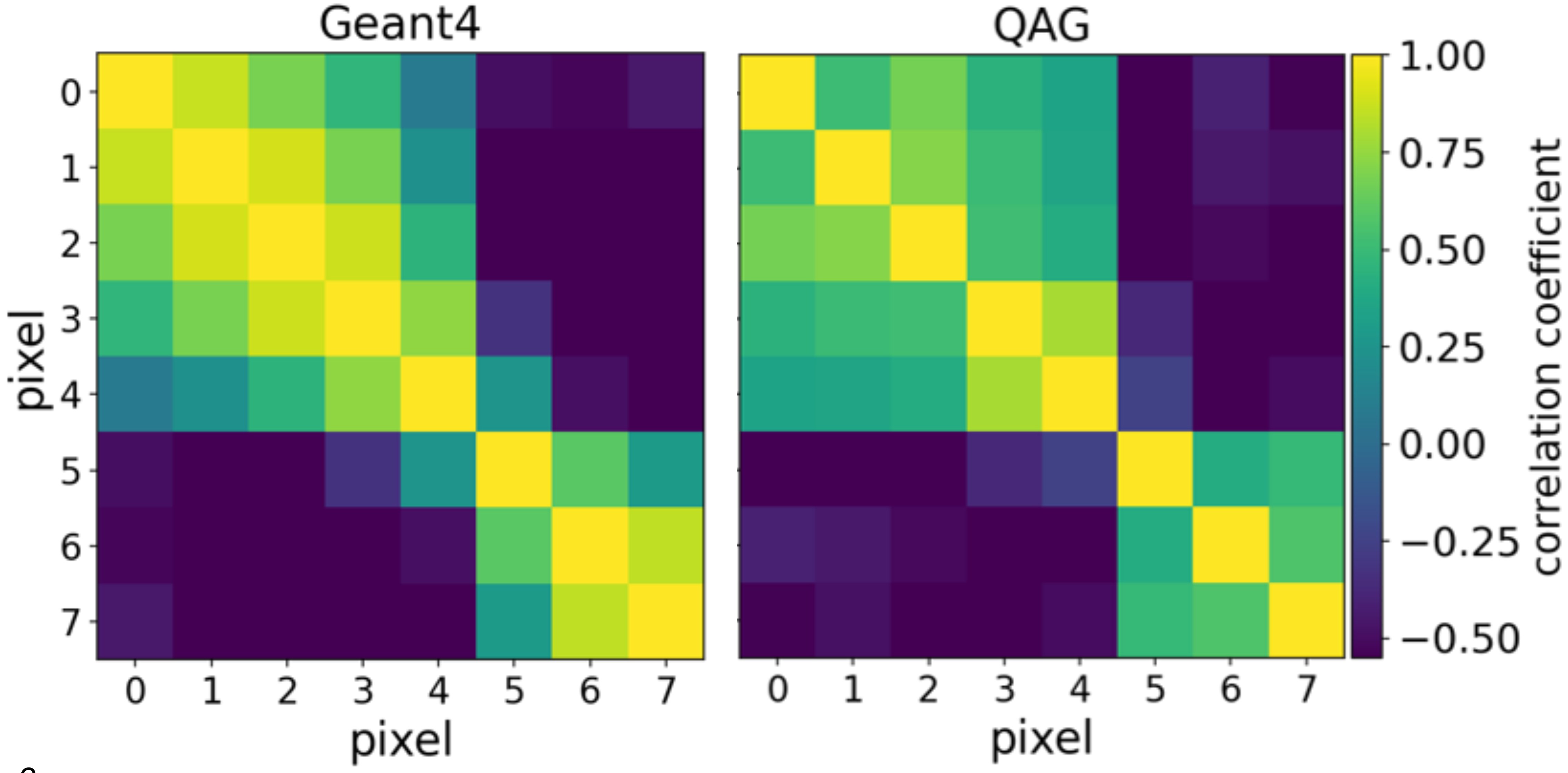
k-means Clusters



Energy Sum



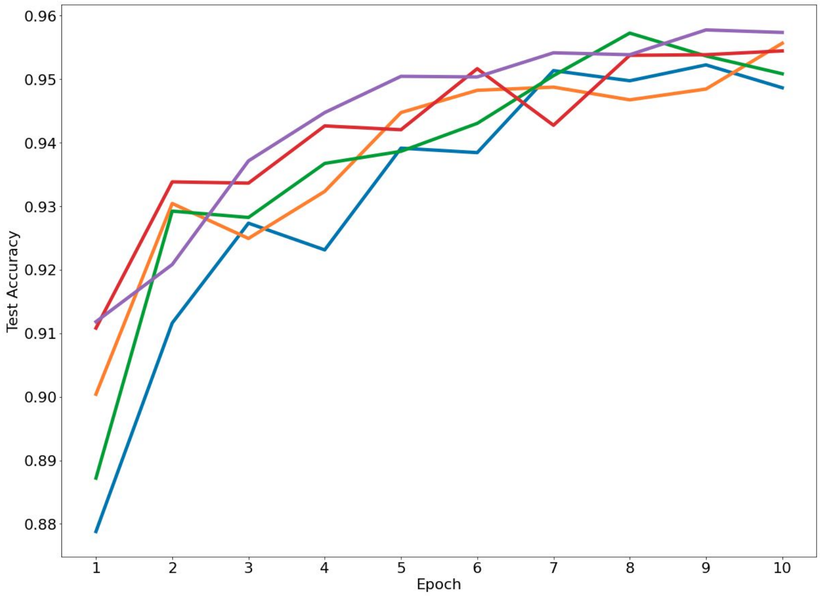
Pixel-wise correlation



Valle Varo
valle.varo@desy.de

How Did We Do?

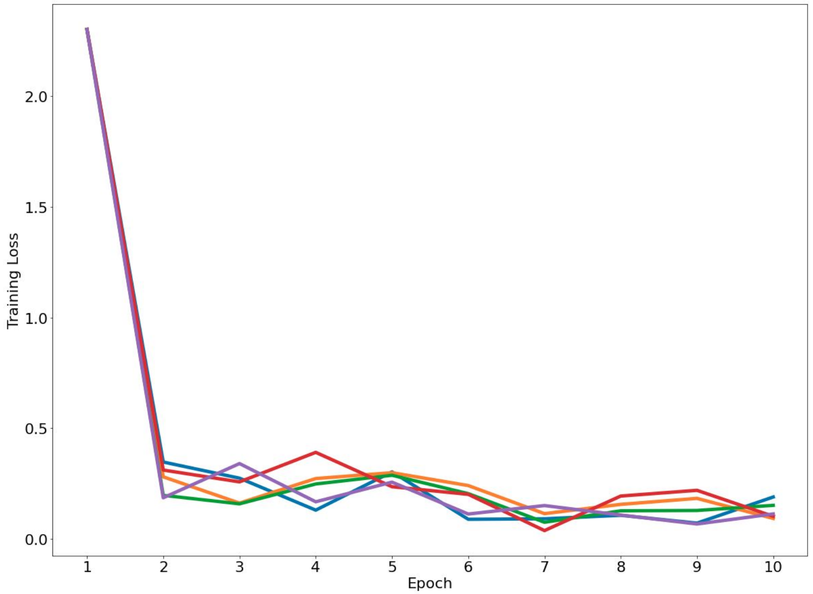
Test Accuracy



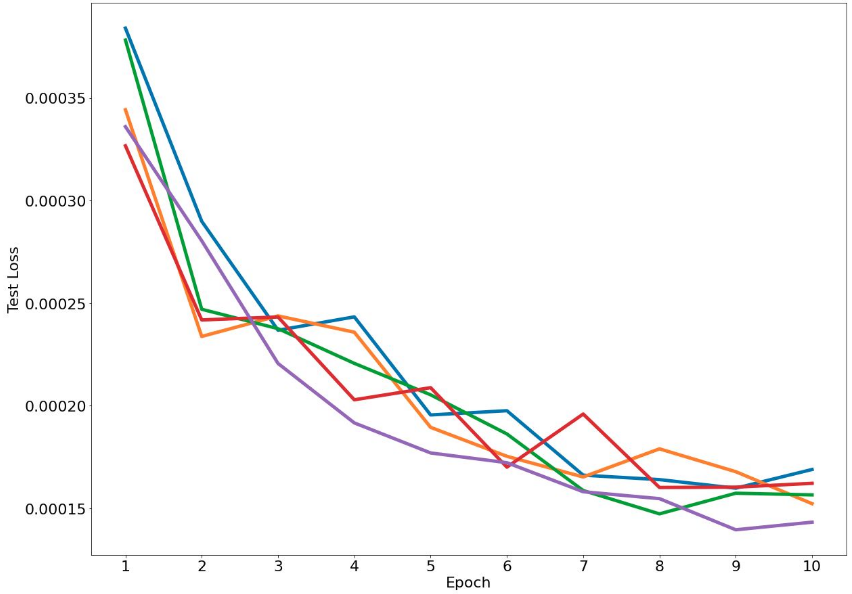
Classical ViT
QViT With π Encoding
QViT With π Encoding And No Rescaling
QViT With $\frac{\pi}{2}$ Encoding
QViT With $\frac{\pi}{2}$ Encoding And No Rescaling

Encoding: scaling the inputs to map between ± value
Rescaling: rescaling the Q, K, V portions to classical shape/dimensions

Training Loss

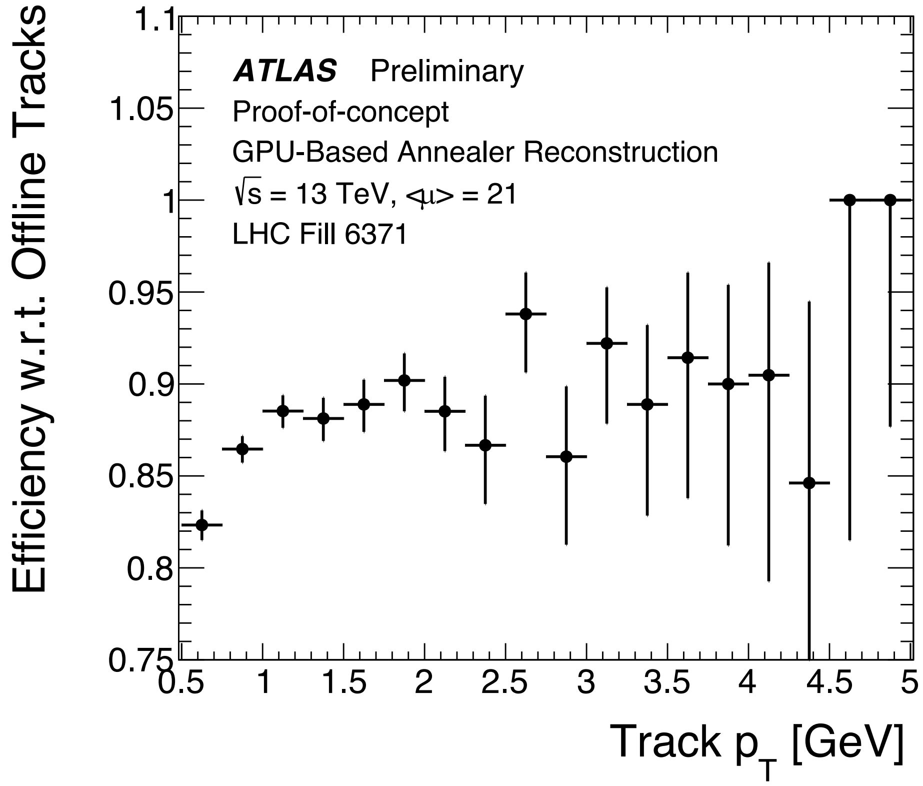


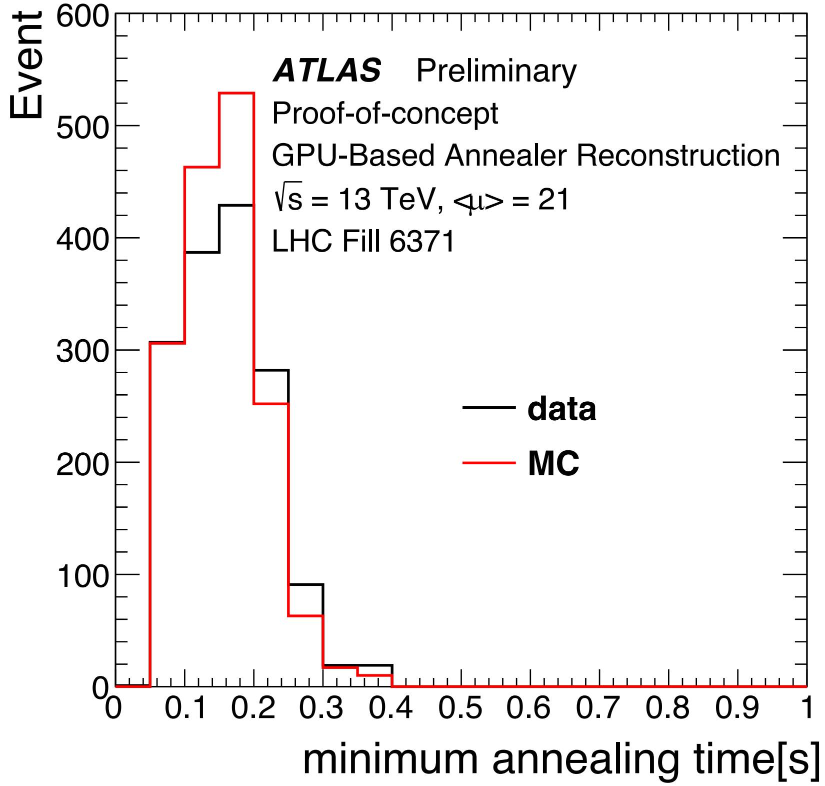
Test Loss



Results with real data

- We applied this algorithm to real ATLAS data taken by non-physics random triggers.
- The efficiency is calculated w.r.t. the ATLAS offline tracks. The matching to the offline tracks is performed if reconstructed tracks with annealing machines share more than 50% of hits with the offline tracks.
- The annealing time was compared with MC sample(10 pions/event with pile-up 20).



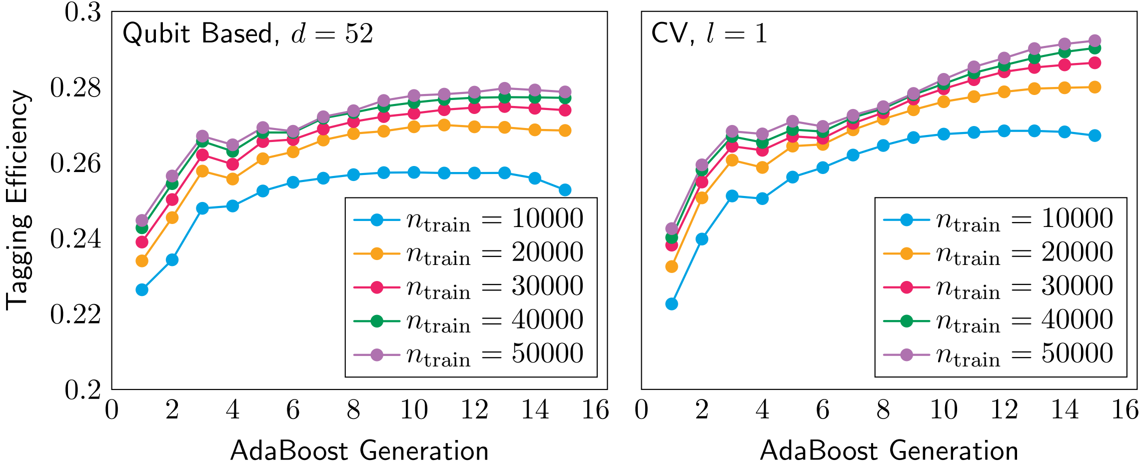


Average pre-processing time for data is ~0.6 sec. (single core, 11th Gen Intel(R) Core(TM) i9-11900K @ 3.50GHz)

- Our algorithm also works successfully with real ATLAS data.
- It is a good starting point to further explore the method.

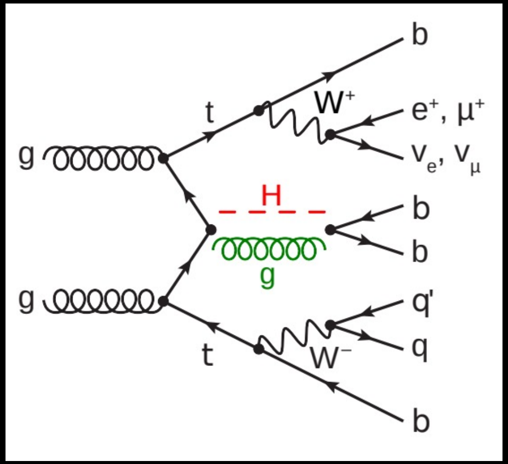
Quantum Support Vector Machines: Results

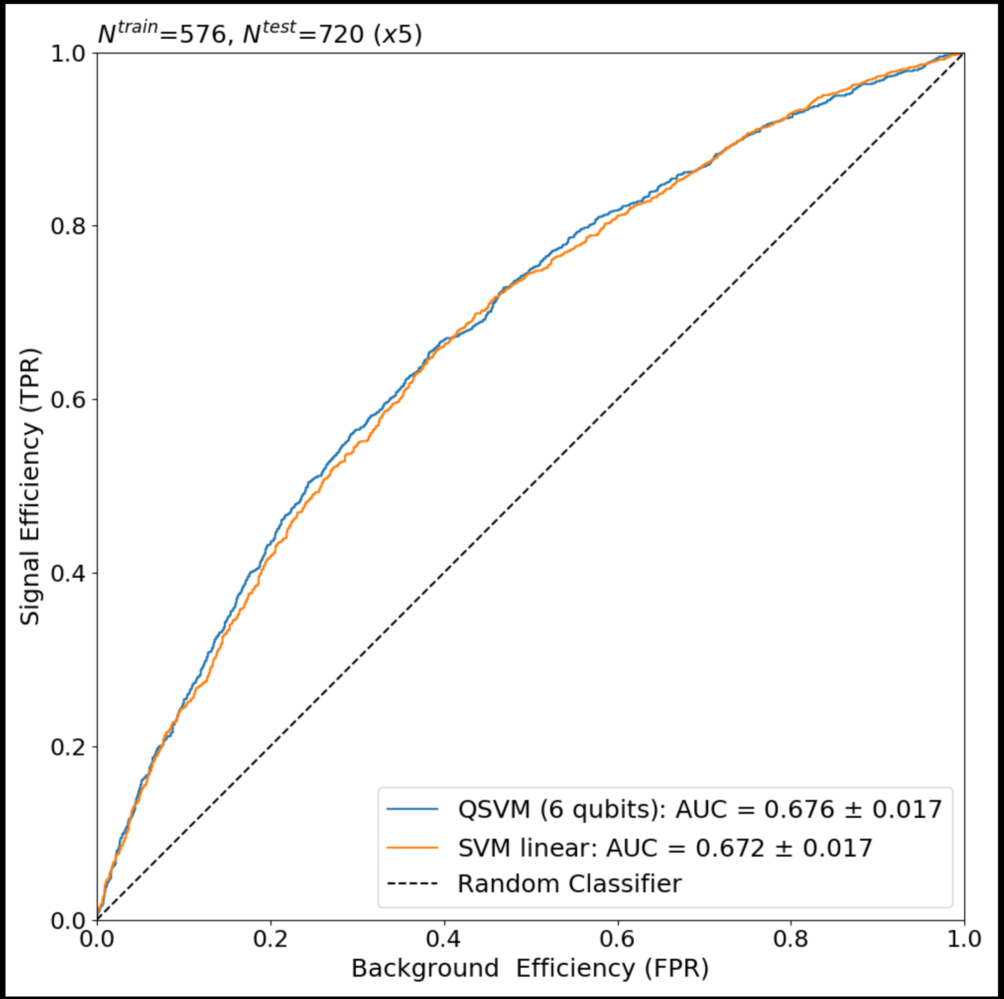
- We construct boosted ensembles of 200 QSVMs.
- Due to simulation constraints, the CV models only include the inital displacement.



Higgs classification

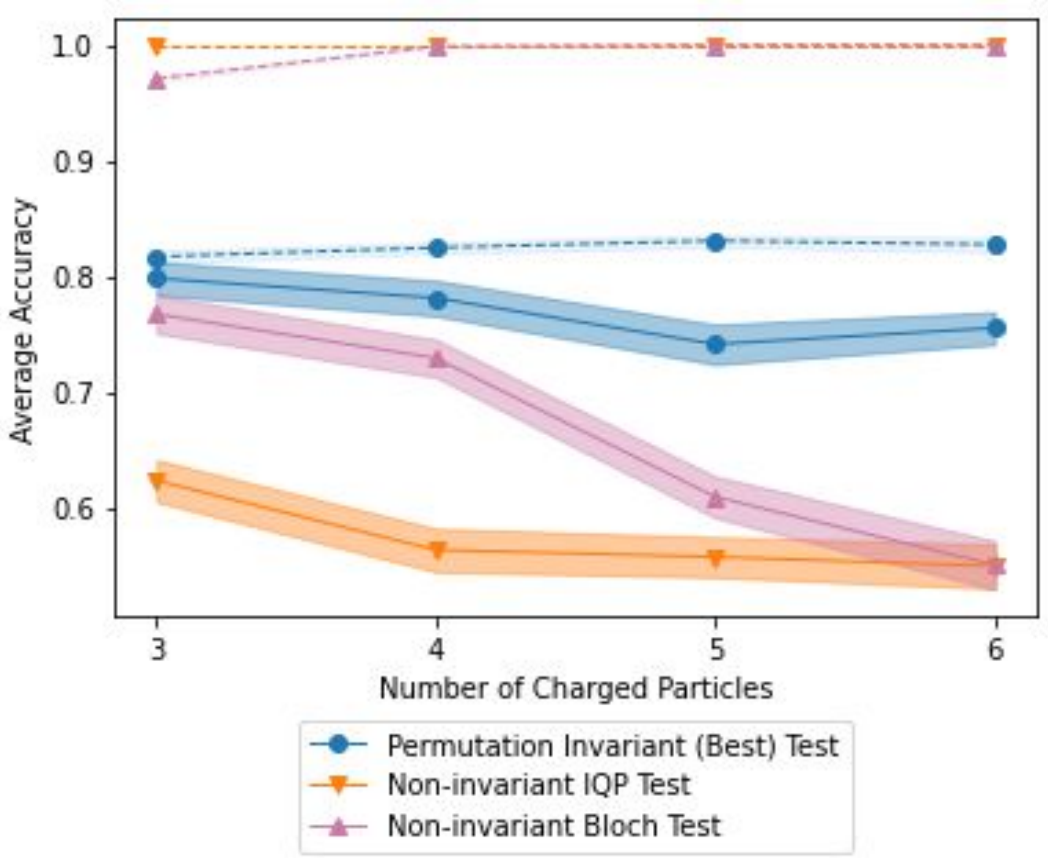
Quantum Support Vector Machine for the *ttH(bb)* event classification[5]





[1] V Belis et al, (2021), *Higgs Analysis with Quantum Classifiers,* EPJ Web Conf

B Meson Continuum Suppression



Weightings in data:

3 particles : 20%

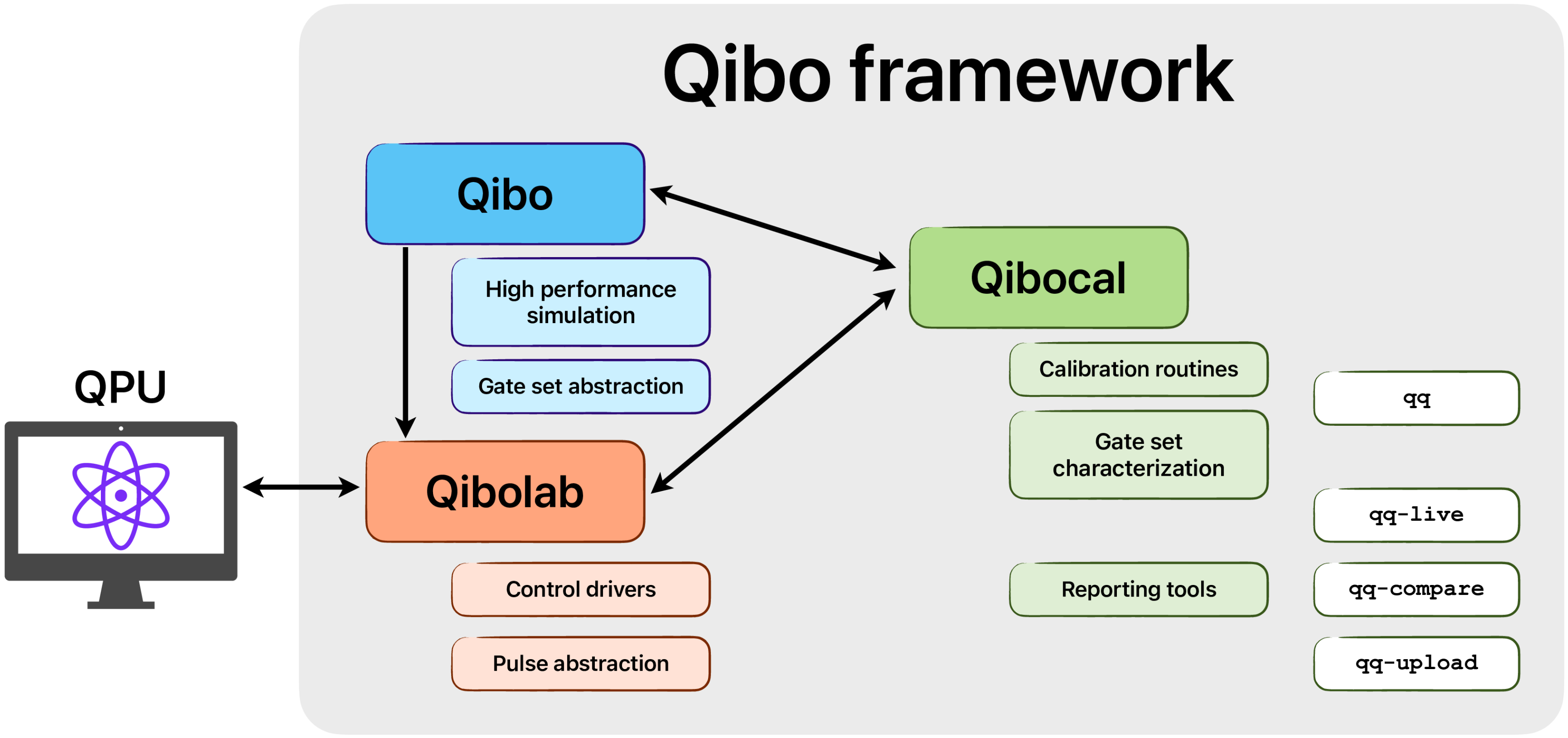
4 particles : 14%

5 particles : 18%

6 particles : 12%

Weighted Averages:

Permutation Invariant = 0.77
Non-invariant = 0.67


Qibo framework
QPU
Qibo
High performance simulation
Gate set abstraction
Qibolab
Control drivers
Pulse abstraction
Qibocal
Calibration routines
Gate set characterization
Reporting tools
qq
qq-live
qq-compare
qq-upload

HEPCloud-Rigetti Pre-Production (Aug ’22)