Reconstruction R&D for HEP in the next decade

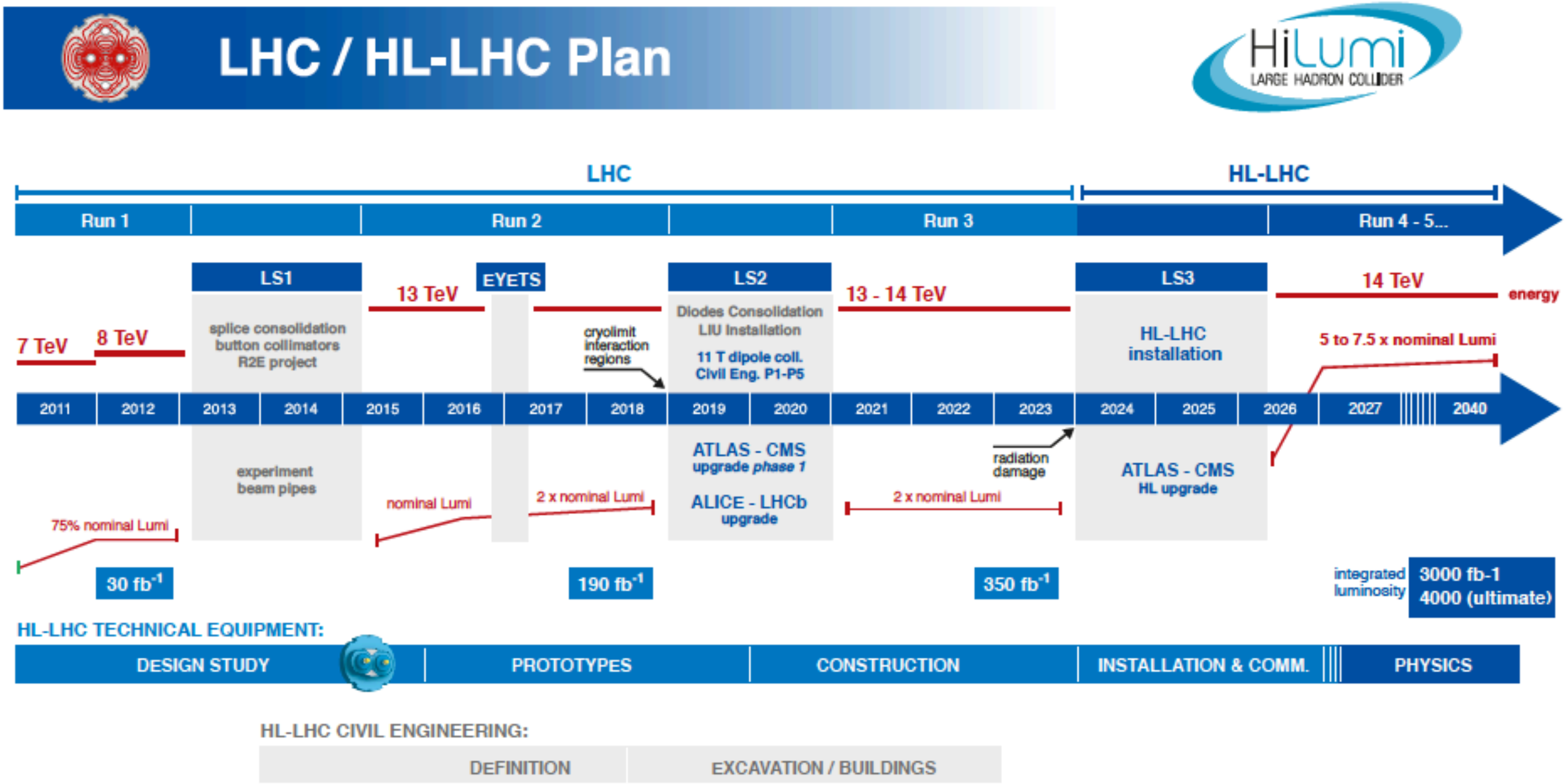
LAWSCHEP 2019 Workshop
November 21, 2019

Slava Krutelyov (UCSD)

using selected progress reports in HSF/IRIS-HEP/CHEP19 and personal perspective

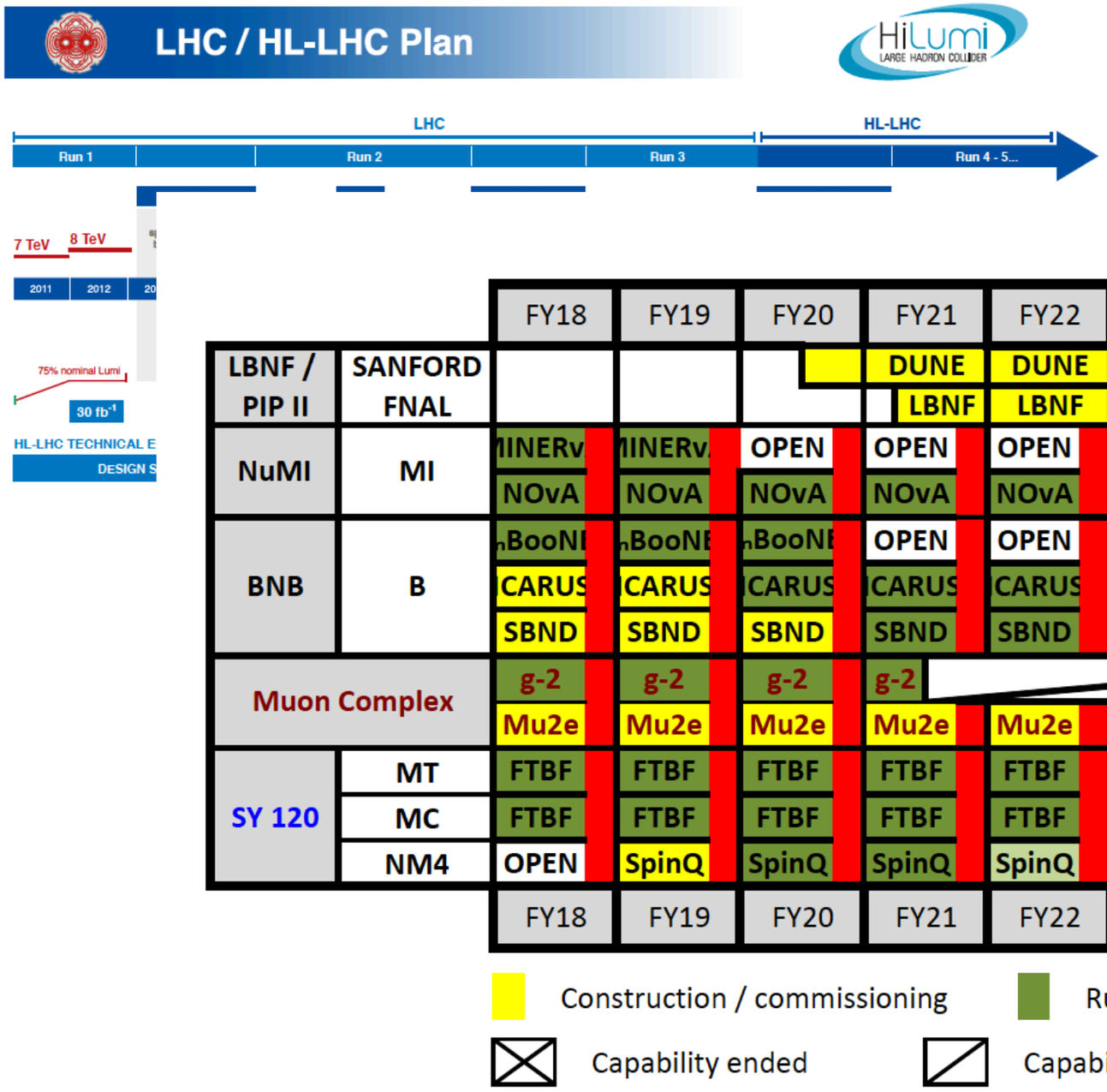
— Computing challenges in HEP
— Changing computing landscape
— Solutions to past challenges and reconstruction areas in depth
— HSF roadmap for 2020s
— Contributions are welcome
— Digest of solutions [see more in the backup]
— Summary

Leading facilities in HEP: CERN



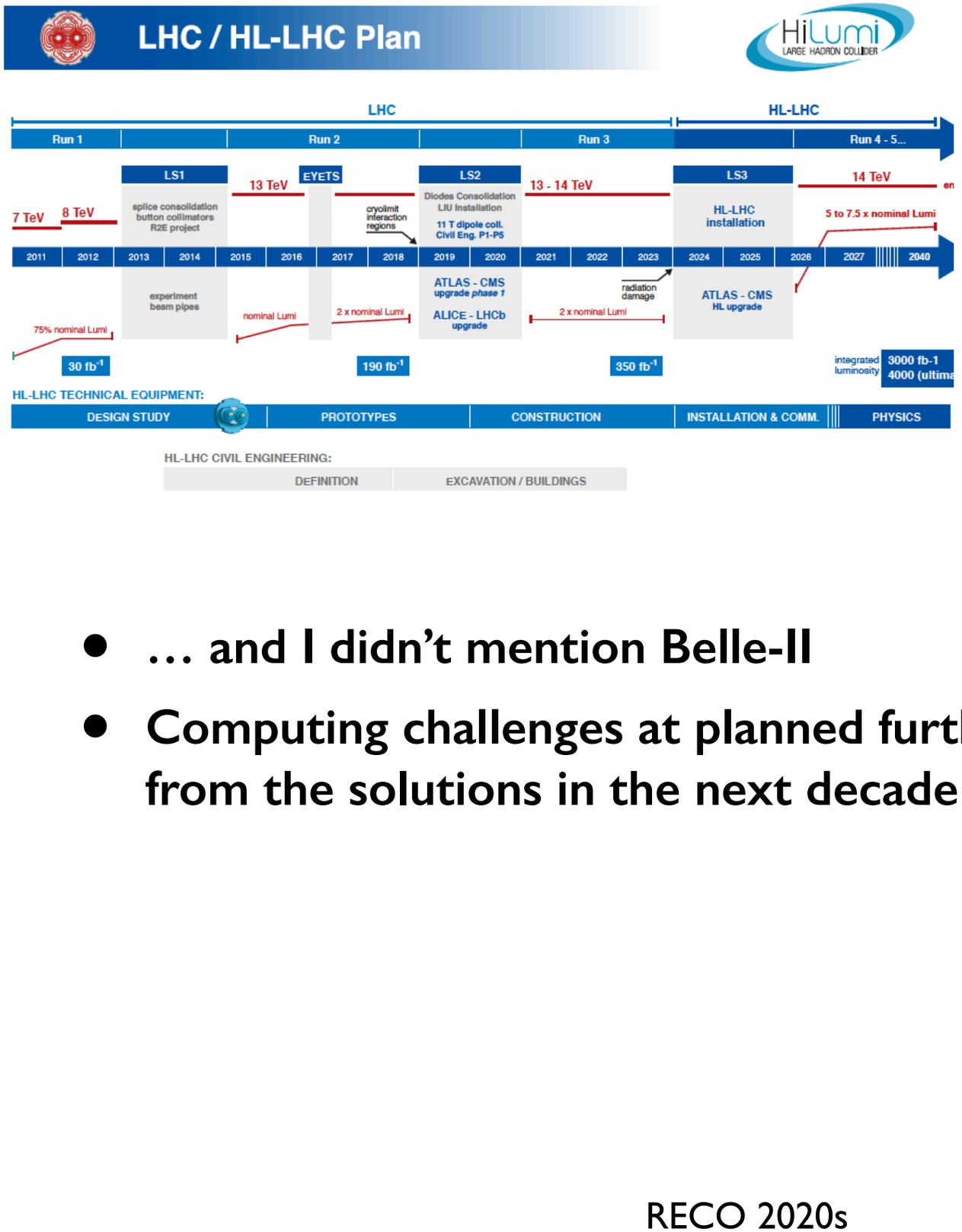
- **Schedule last updated Oct 2019.**

Leading facilities in HEP: Fermilab



- **Schedule last updated July 2019.**

Leading facilities in HEP



Office of the CRO July 2019

DRAFT LONG-RANGE PLAN

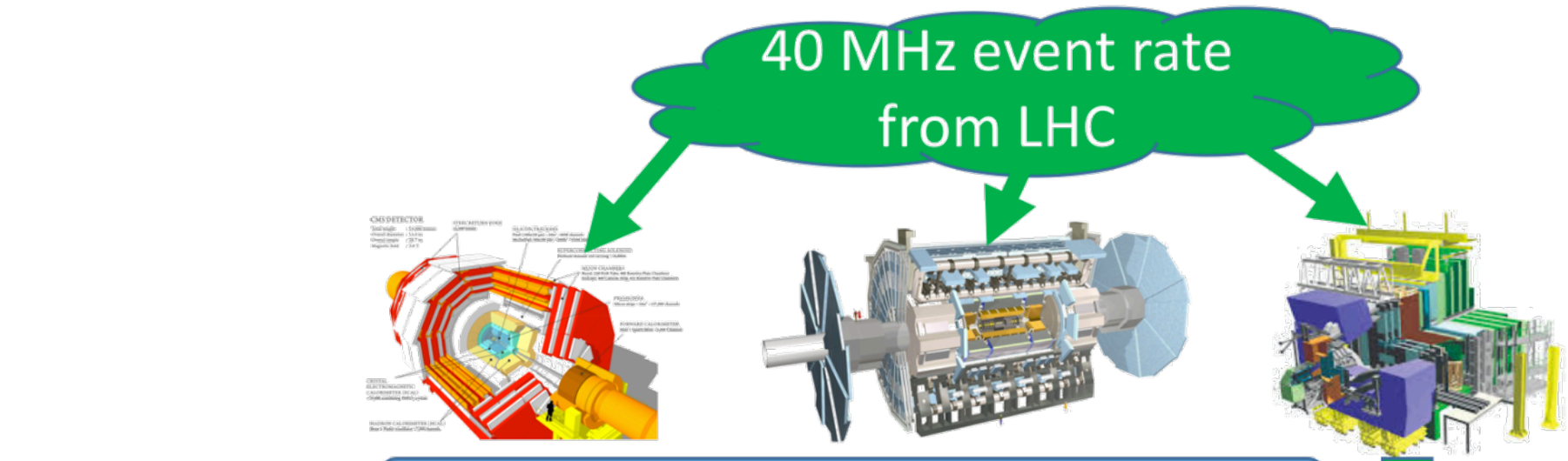
LBNF / PIP II	SANFORD FNAL	FY18	FY19	FY20	FY21	FY22	FY23	FY24	FY25	FY26	FY27	FY28	FY29	FY30	
					DUNE	DUNE	DUNE	DUNE	DUNE	DUNE	DUNE	DUNE	DUNE	DUNE	n
					LBNF	LBNF	LBNF	LBNF	LBNF	LBNF	LBNF	LBNF	LBNF	LBNF	
NuMI	MI	MINERvA	MINERvA	OPEN	OPEN	OPEN	OPEN	OPEN	OPEN						
		NOvA	NOvA	NOvA	NOvA	NOvA	NOvA	NOvA	NOvA						
BNB	B	MicroBooNE	MicroBooNE	MicroBooNE	OPEN	OPEN	OPEN	OPEN	OPEN	LONG SHUTDOWN AND FACILITY COMMISSIONING		OPEN	OPEN	OPEN	
		CARUS	CARUS	CARUS	CARUS	CARUS	CARUS	OPEN	OPEN			OPEN	OPEN	OPEN	
		SBND	SBND	SBND	SBND	SBND	SBND	OPEN	OPEN			OPEN	OPEN	OPEN	
Muon Complex		g-2	g-2	g-2	g-2									OPEN	m
		Mu2e	Mu2e	Mu2e	Mu2e	Mu2e	Mu2e	Mu2e	Mu2e			Mu2e	Mu2e	OPEN	
SY 120	MT	FTBF	FTBF	FTBF	FTBF	FTBF	FTBF	FTBF				FTBF	FTBF	FTBF	p
	MC	FTBF	FTBF	FTBF	FTBF	FTBF	FTBF	FTBF				FTBF	FTBF	FTBF	
	NM4	OPEN	SpinQ	SpinQ	SpinQ	SpinQ	OPEN	OPEN				OPEN	OPEN	OPEN	
		FY18	FY19	FY20	FY21	FY22	FY23	FY24	FY25	FY26	FY27	FY28	FY29	FY30	

Construction / commissioning Run Subject to PAC review Shutdown

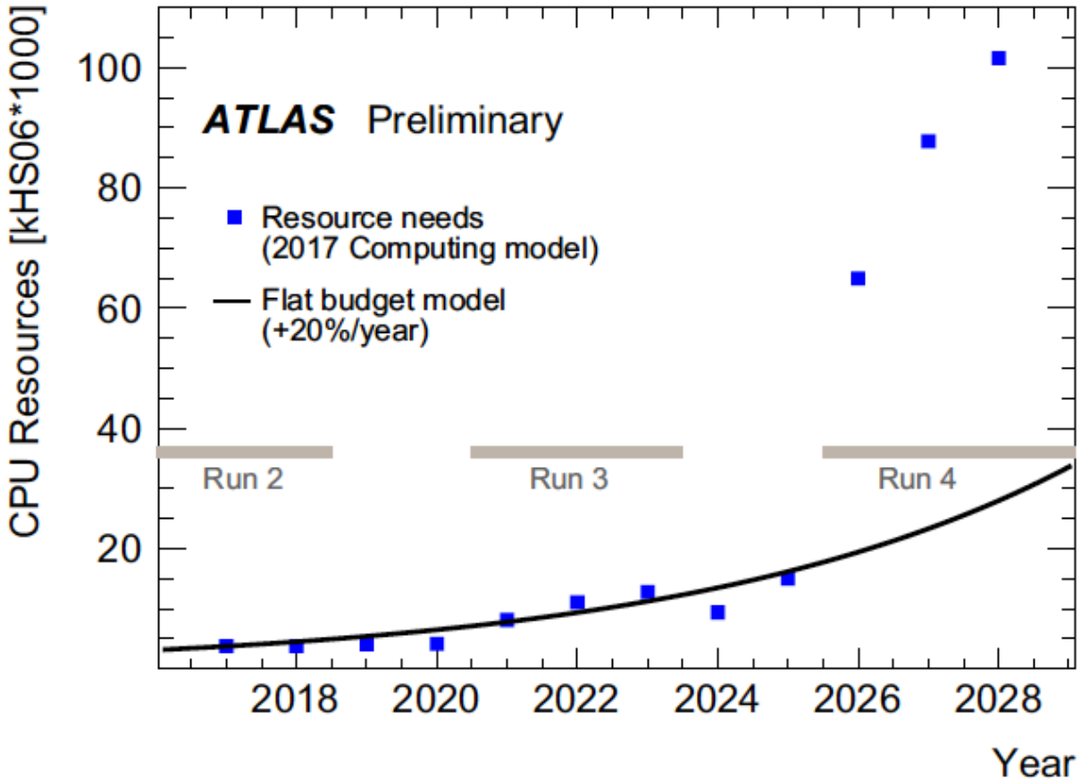
Capability ended Capability unavailable

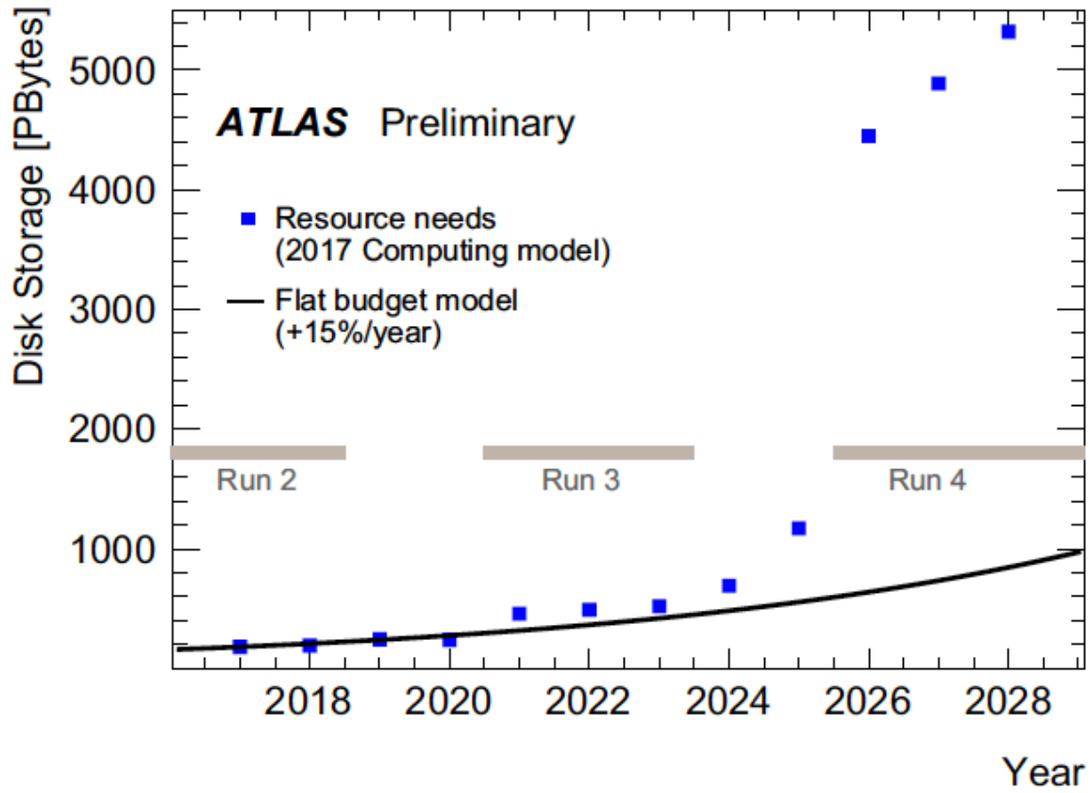
- **… and I didn't mention Belle-II**
- **Computing challenges at planned further future facilities (e.g. FCChh) will take from the solutions in the next decade**

HL-LHC requires more computing



Computing challenges at leading facilities





- **Projections for both CPU and disk needs exceed expected costs assuming flat budget model**
- **As of HSF CWP, we are factor of ~5 too short**
- **Similar projections available for CMS**

DUNE

Scaling https://indico.cern.ch/event/773049/contributions/3581360

2018: ProtoDUNE event
6 APA ~ 130 MB
At 25 Hz

2025: Beam/cosmic ray event
in 1 FD module -- 150 APA ~ 6GB at < 0.1 Hz

Someday: Supernova
150x4x20,000 5 ms APA
~400 TB. 1/month

CHEP 2019

Oregon State University

DUNE 31

- **Large data volumes expected**
  - ✓ Still, comparable to current (Run2/Run3) use at the ATLAS/CMS experiments

Improved estimates for HL-LHC

European Strategy Update

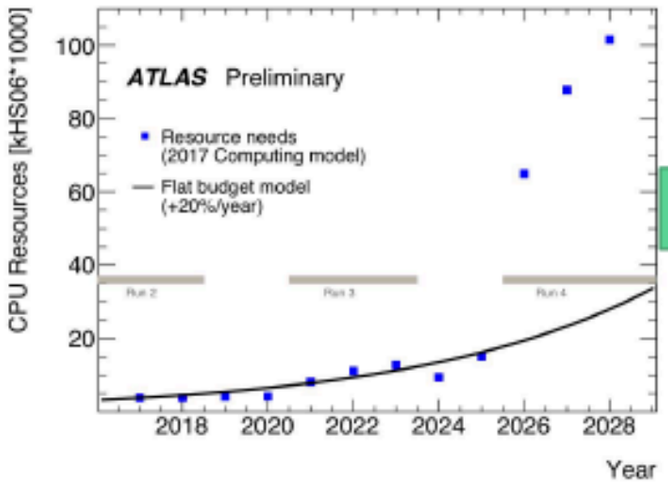
Instrumentation and Computing

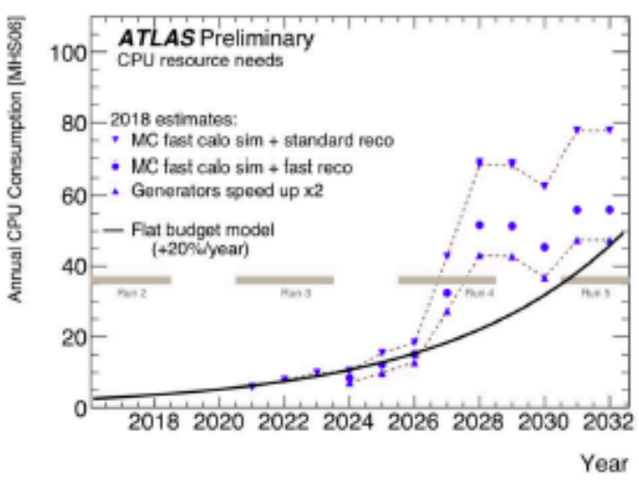
Xinchou Lou, Brigitte Vachon
Scientific Secretaries: Emilia Leogrande, Rogers Jones

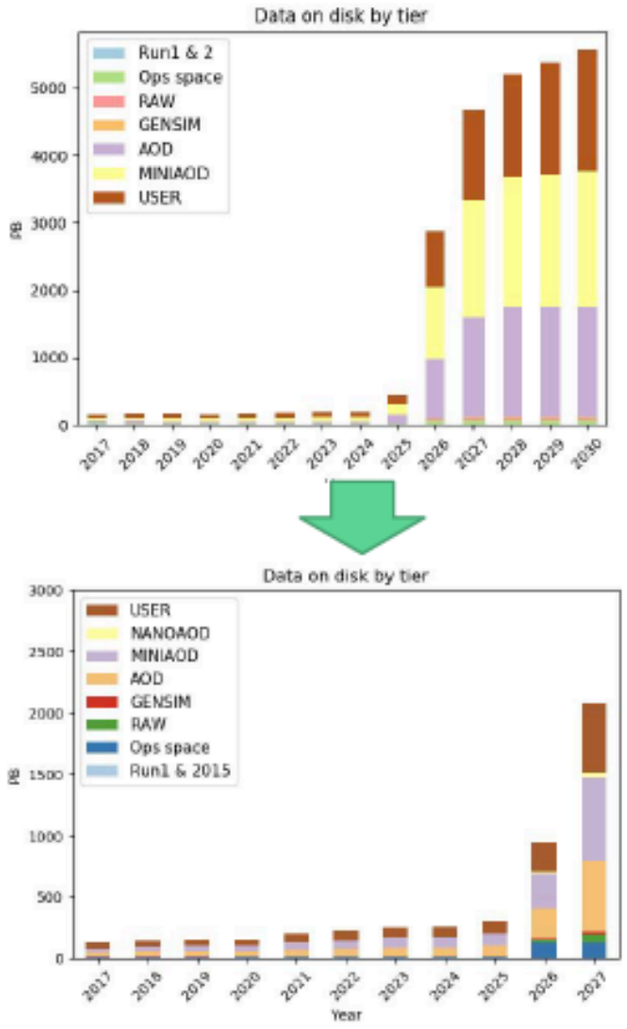
Oct 2019 HSF-JENAS https://indico.cern.ch/event/852242/contributions/3583914

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





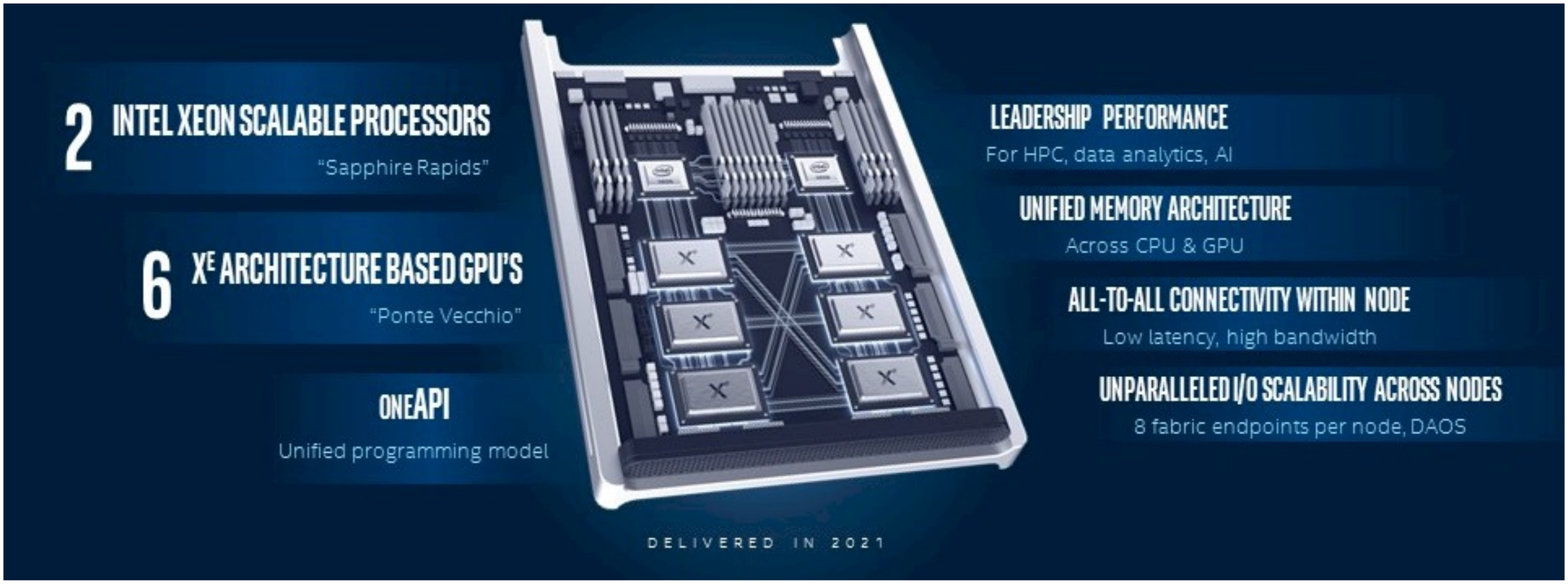


- **Up to a factor of two reduction in resource needs with new estimates.**
- **This is still a factor of 3-5 too short from the flat budget.**

Computing landscape changes

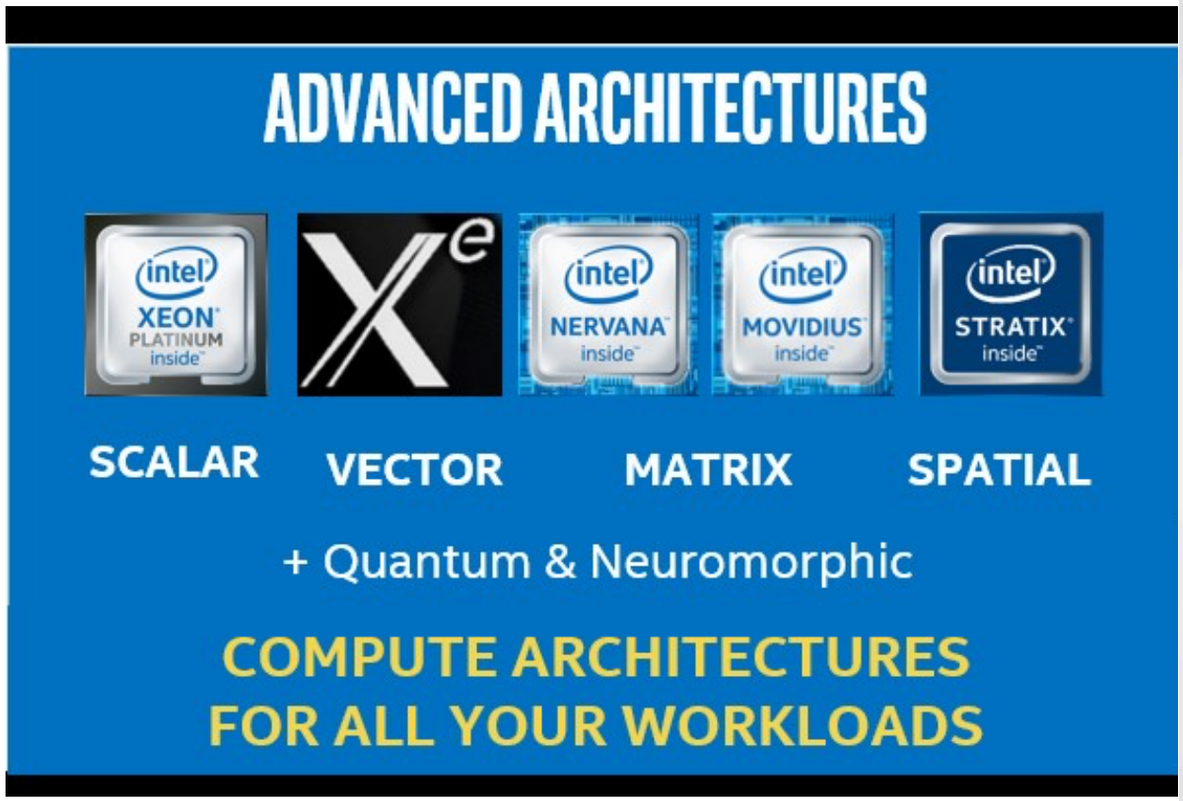
- **CPU estimates for HL-LHC are still based on the classical CPU-only computing**
- **This landscape changes, driven by lower cost per operation on GPUs, FPGAs and similar compute hardware matched with cost-productive applications**
- **AI/ML workloads are perhaps the leading general interest area that drive these developments**
- **Science funding agencies are motivated to follow suit**
- **Future science HPCs are expected to have accelerators dominate the total "per node" compute capacity**

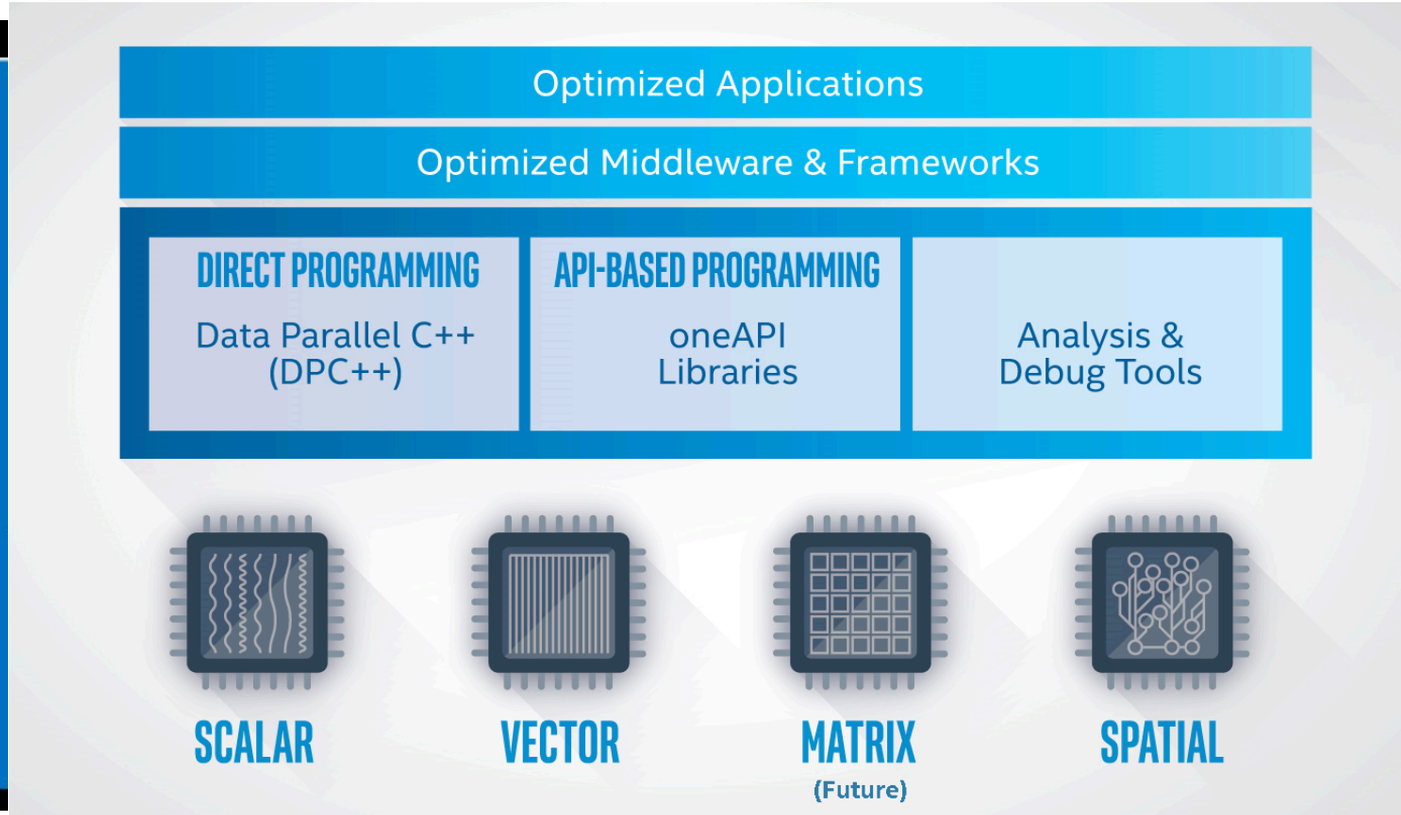
Example near-future HPC



- Aurora "A21" machine to be delivered in 2021. One of the first exascale HPCs
- Expect about 90% of FP compute capacity from the GPUs
  - More thoughts or speculations in https://www.nextplatform.com/2019/11/20/doing-the-math-on-future-exascale-supercomputers/
- Early Science Program already includes LHC (ATLAS)
  - https://press3.mcs.anl.gov/aurora/

This will likely get more complex





- **Can expect similar activities from other industry leaders**
- **Heterogeneous computing is becoming a reality**
- **Specialized software solutions can work but portability will be required to make use of different available resources**
- **Portable software solutions are coming with support/funding from industry and from science funding organizations**
  - **KOKKOS, ALPACA, SYCL**
  - **OneAPI (Intel, beta release last week) based upon the SYCL* Specification**

Inspirational: large speedups are possible

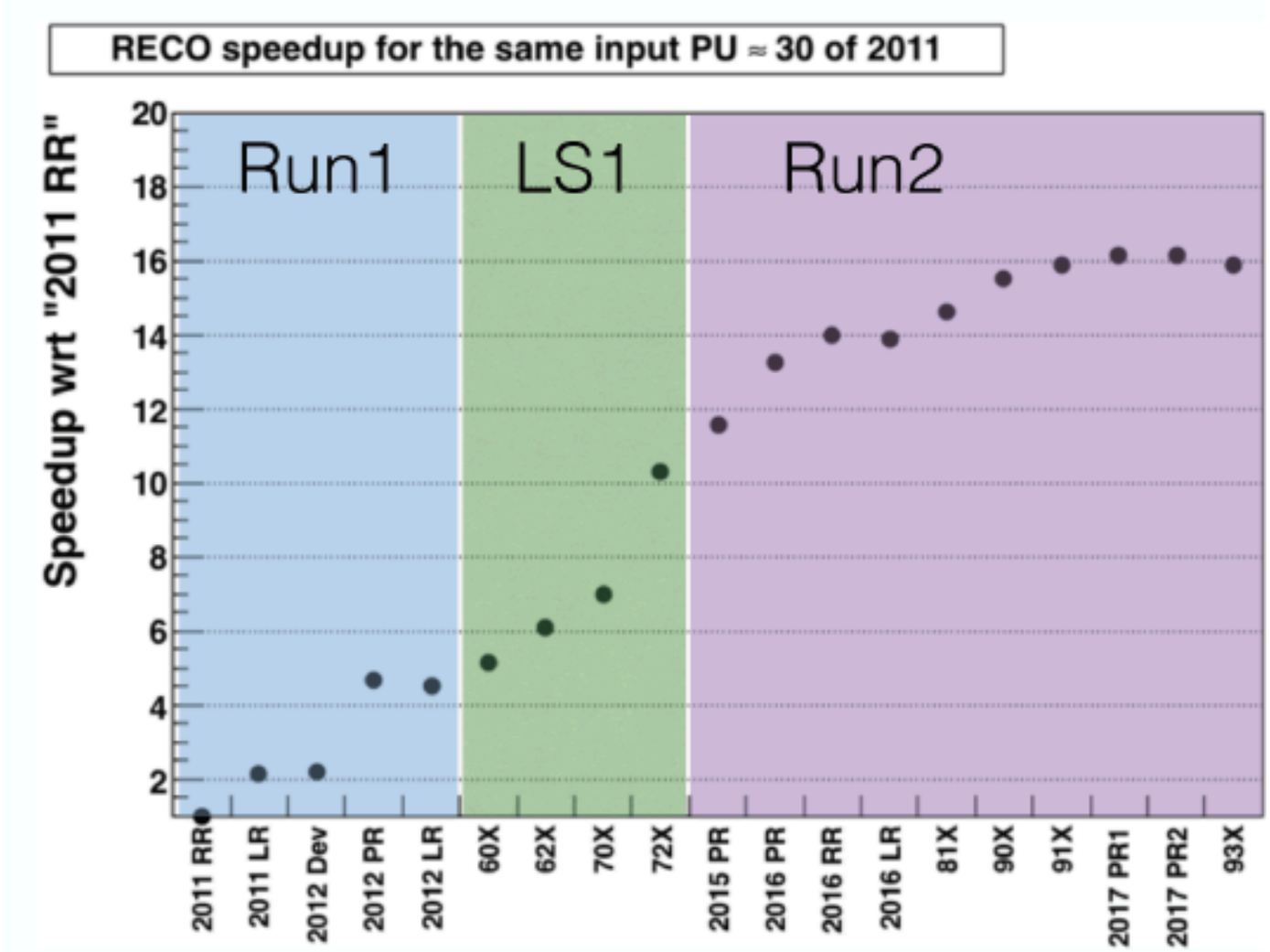
pp reco: historical perspective

- Reconstruction algorithms evolve through the years. Driven by novel algorithms and technical software optimizations and innovations.

➡ 6 years in one plot

- ✓ Same data input
- ✓ Same hardware to test
- ✓ Different releases
  - ✶ includes new algorithms, modifications to existing algorithms, and technical improvements

➡ Same data reconstruction is faster by x16



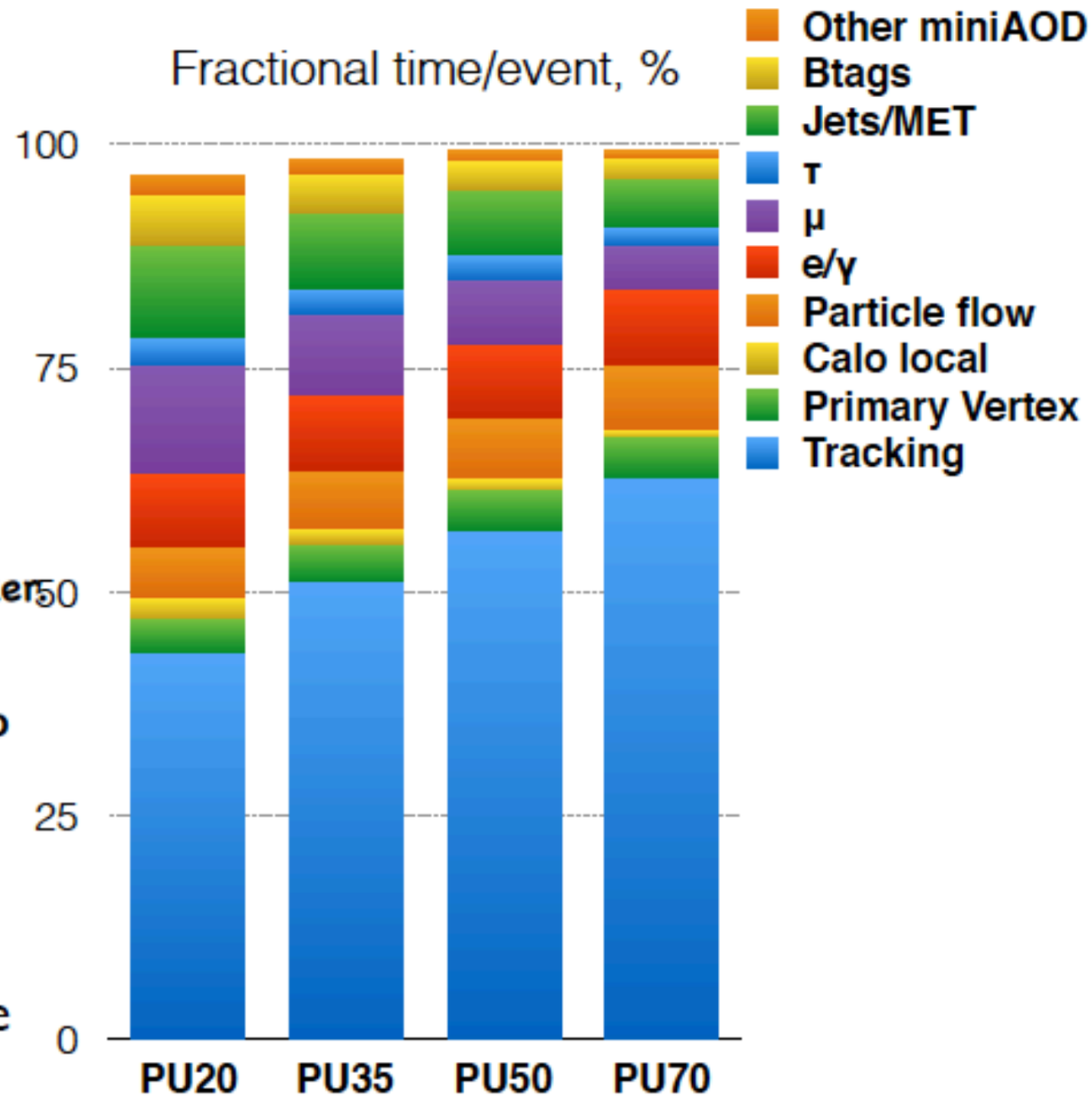
PR is prompt reconstruction
RR is (end-of-year) rereconstruction
LR is legacy reconstruction
releases ending in X are development

https://indico.cern.ch/event/587955/contributions/2937619

CMS pp RECO "current" state of Run2

pp Run-2 reco: in depth

- Look at time spent by one reco application, starting from RAW producing miniAOD
- CPU use by category (exclusive)
  - ➡ Notes
    - e/γ reco includes specialized electron/conversion tracking
    - about 40-60% of tracking is in iterations seeded in the outer tracker. This fraction grows with pileup.
  - ➡ Tracking fraction grows with pileup
  - ➡ Most other categories are roughly similar as a function of pileup



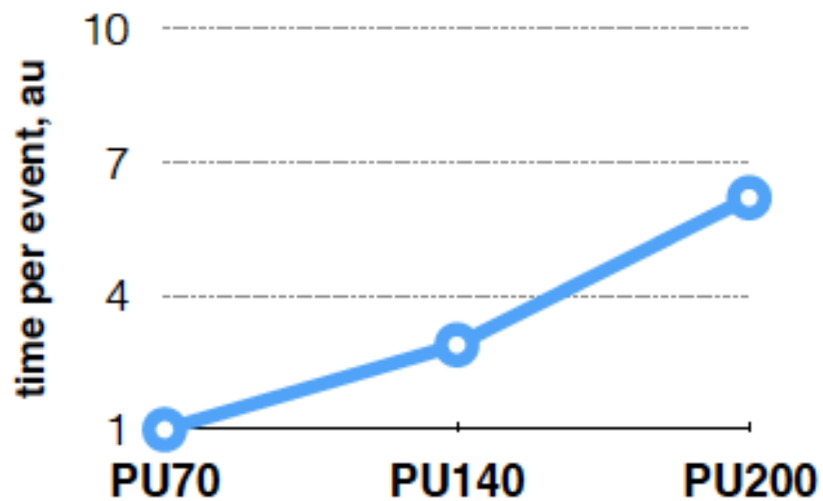
- It's fair to expect that Run-3 will be similar

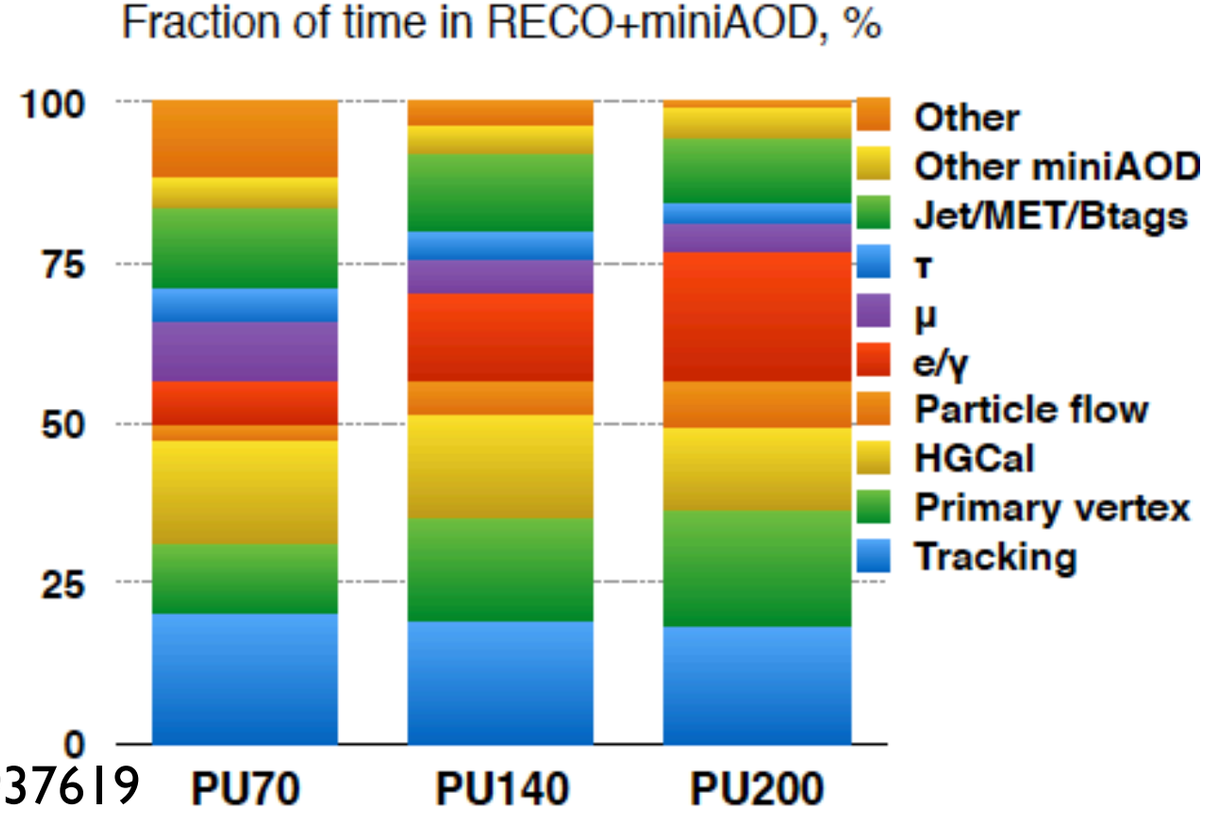
https://indico.cern.ch/event/587955/contributions/2937619

CMS pp RECO "current" state for HL-LHC

pp Phase-2/HL-LHC reco: in depth

- Look at time spent by one reco application, starting from RAW producing miniAOD
- CPU use by category (exclusive)
  - ➡ Notes
    - tracking relies only on the inner pixel tracker iterations
    - primary vertex includes 4D reco, including track timing information
    - e/γ reco includes specialized electron/conversion tracking currently partly duplicated with alternative HGCAL cluster inputs
  - ➡ Most categories are roughly similar as a function of pileup
- This picture is expected to evolve in the next 5 or so years





https://indico.cern.ch/event/587955/contributions/2937619

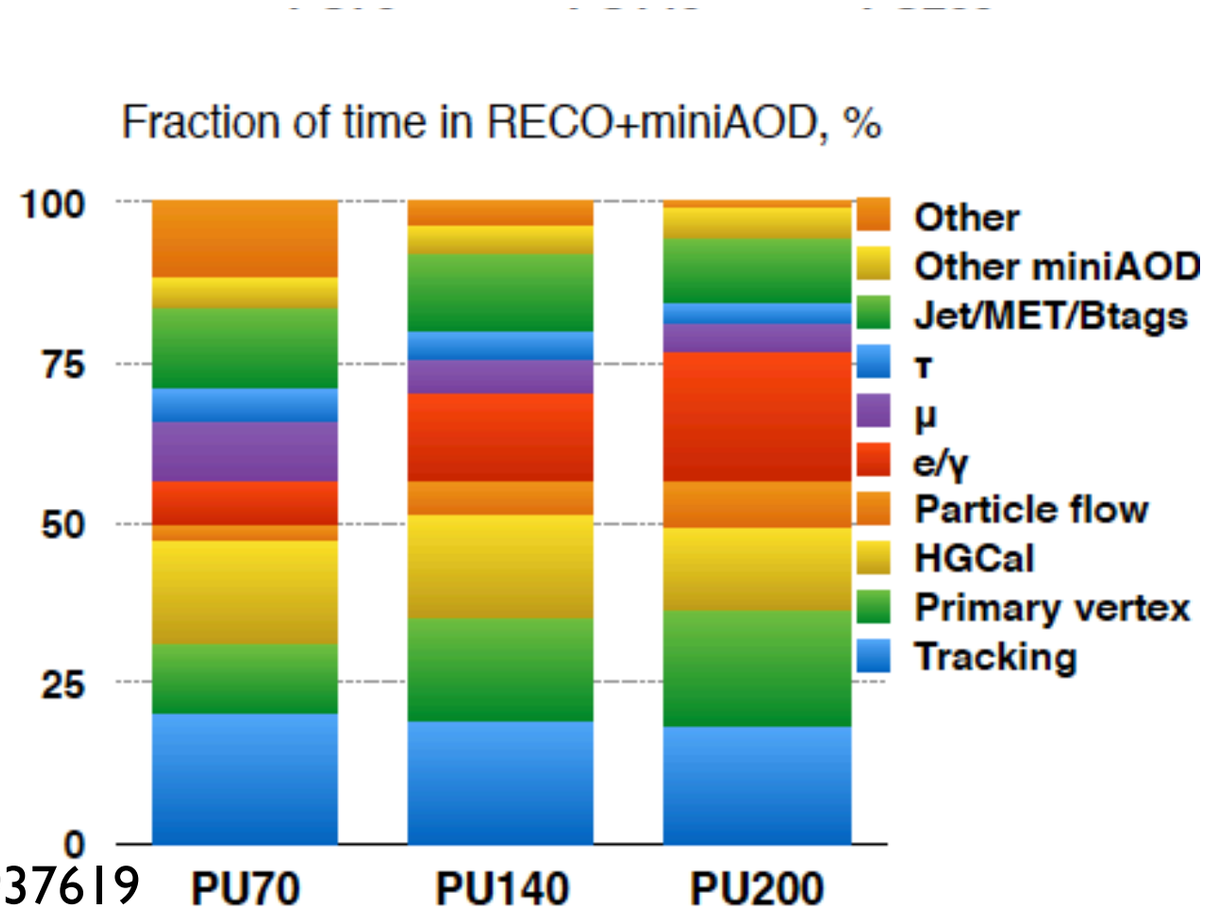
CMS pp RECO "current" state for HL-LHC

- **The balance of contributions is quite likely to change**
  - **Only tracking here is more optimized for HL-LHC, perhaps even too restrictive**
- **Major contributors indicate the areas that need more focus in the coming years**
- **Significant progress was made since early 2018 release when the plot was made, to be soon deployed for for improved cost estimates**

- tracking relies only on the inner pixel tracker iterations
- primary vertex includes 4D reco, including track timing information
- e/γ reco includes specialized electron/conversion tracking currently partly duplicated with alternative HGCAL cluster inputs

➡ Most categories are roughly similar as a function of pileup

- This picture is expected to evolve in the next 5 or so years



https://indico.cern.ch/event/587955/contributions/2937619

HSF roadmap: reconstruction

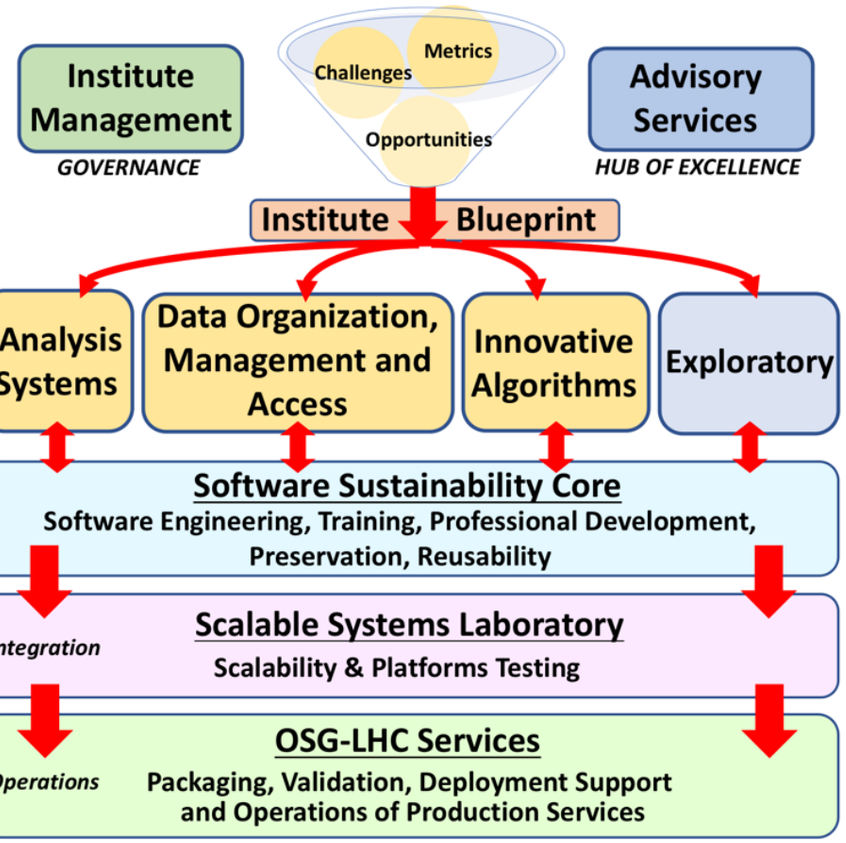
- **Seven key areas were identified to exploit the full power of the enormous datasets that we will be collecting**
  - ✳ **Three areas concern the increasingly parallel and heterogeneous computing architectures**
  - ✓ Evolve current toolkit and algorithm implementations, and best programming techniques, to better use SIMD capabilities of current and future CPU architectures
  - ✓ Adopt programming techniques, to improve the throughput of multithreaded software trigger and event reconstruction applications
  - ✓ Use computing architectures with technologies beyond CPUs. Assess and minimize possible additional costs coming from the maintenance of multiple implementations.
  - ✓ Enable the development, automation, and deployment of extended QA and QC tools and facilities for software trigger and event reconstruction algorithms.
  - ✓ Real-time analysis techniques: evaluate and demonstrate the tools needed … for real-time detector calibration and validation that enable full offline analysis chains to be ported into real-time, and frameworks that allow non-expert offline analysts
  - ✓ Develop and demonstrate efficient techniques for physics object reconstruction and identification in complex environments similar or higher than the HL-LHC.
  - ✓ Evolve or rewrite toolkits/algorithms limiting processing at high pileup; most significantly charged-particle tracking. Deploy new algorithms, including advanced machine learning techniques.

How to contribute

- **Naturally, a large fraction of work is done in the context of a specific experimental collaboration**
- **Community effort via HSF and national institutes**
- **HSF Reconstruction and Software Triggers group**
  - ✓ https://hepsoftwarefoundation.org/workinggroups/recotrigger.html
  - ✓ Annual large meetings help following the activities
  - ✓ 18-22 March 2019 “HOW” workshow, joint HSF/OSG/WLCG
    - ✳ https://indico.cern.ch/event/759388
  - ✓ 11-15 May 2020 joint HSF/WLCG workshop (announced earlier today)
    - ✳ https://indico.cern.ch/event/860332/
- **IRIS-HEP Innovative Algorithms**
  - ✓ https://iris-hep.org/ia.html

How to contribute

- **Naturally, a large fraction of work is done in the context of a specific experimental collaboration**
- **Community effort via HSF and national institutes**
- **HSF Reconstruction and Software Triggers group**
  - ✓ https://hepsoftwarefoundation.org/workinggroups/recotrigger.html
- **IRIS-HEP Innovative Algorithms**
  - ✓ https://iris-hep.org/ia.html

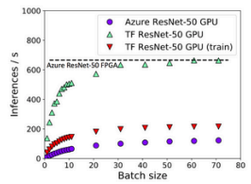


ACTS

Development of experiment-independent, inherently parallel track reconstruction.
More information

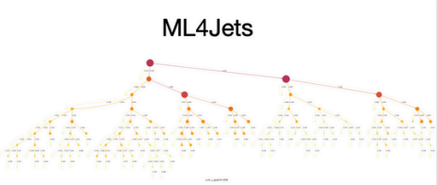
FastPID

Fast PID simulation for LHCb
More information



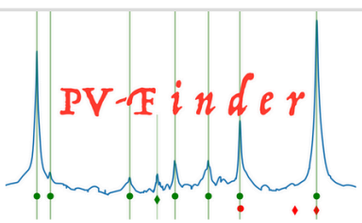
ML on FPGAs

Fast inference of deep neural networks on FPGAs
More information



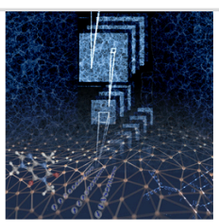
Machine Learning for jets

Machine learning for jets
More information



PV-Finder

CNNs to find primary vertices
More information



exploratory-ml

Analysis Reinterpretation
More information

Digest of topics

- **In the following, I pick a few topics, more can be found in the backup or on the HSF and IRIS-HEP web sites**

- **CMS use of GPU in high level trigger (PATATRACK project activity): inner pixel tracker track reconstruction and calorimeter energy pulse reconstruction**
  - ✓ See backup for GPU-CPU implementations on ALICE and LHCb
  - ✓ Here there is a significant effort towards portability

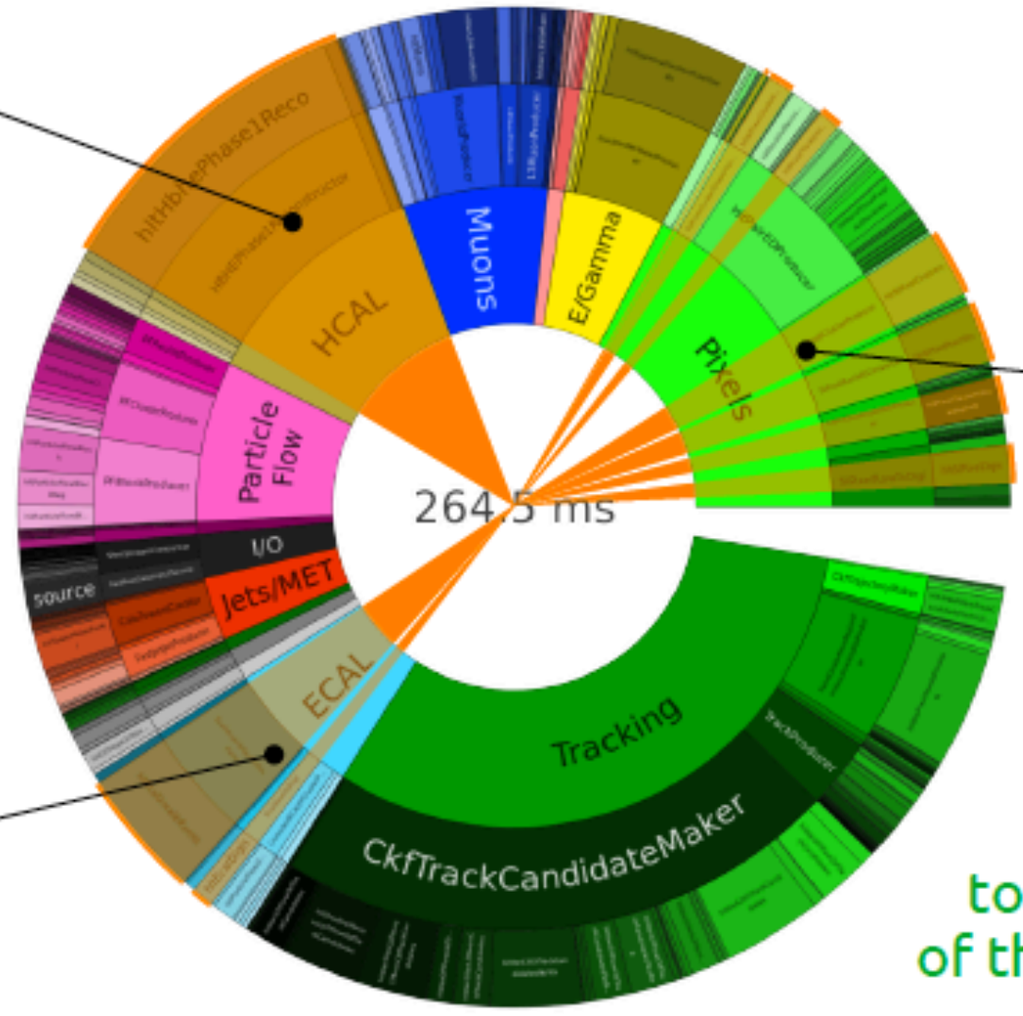
- **Parallelized/vectorized combinatorial Kalman filter based tracking (mkFit project)**
  - ✓ Generic concept with a working demonstration of Run-2/3 CMS tracker

- **ML on FPGAs (HLS4ML and ML as a service)**

Heterogeneous HLT CMS

offloading to GPUs

HCAL: local reconstruction and calibrations



see Monday's talk in Track 9
"High Performance Computing for High Luminosity LHC"

pixel tracking: global reconstruction
details on the next slides

ECAL: local reconstruction and calibrations

today we can offload ~24% of the online reconstruction !

https://indico.cern.ch/event/773049/contributions/3474336

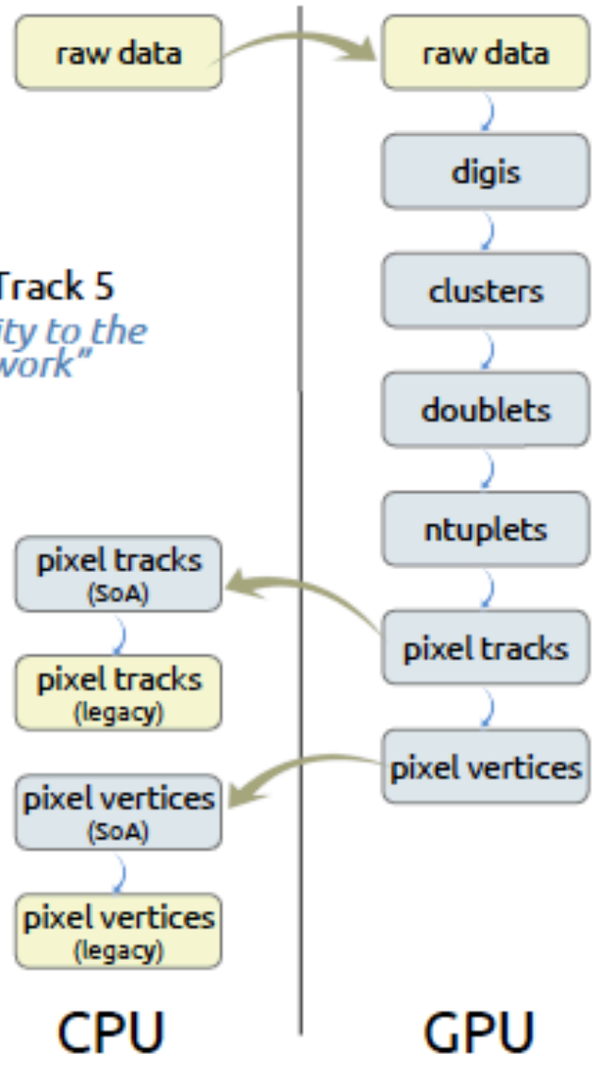
Heterogeneous HLT CMS

the Patatrack demonstrator

- the overall approach
  - reconstruct pixel-based tracks and vertices on the GPU
  - leverage existing support for threads and on-demand reconstruction
  - minimise data transfer

see Monday's talk in Track 5
"Bringing heterogeneity to the CMS software framework"

- the full workflow
  - copy the raw data to the GPU
  - run multiple kernels to perform the various steps
    - decode the raw data
    - cluster the pixel hits
    - form hit doublets
    - form hit ntuplets (triplets or quadruplets) with a Cellular automaton algorithm
    - clean up duplicates
  - take advantage of the GPU computing power to improve the physics
    - fit the track parameters (Riemann fit, broken line fit) and apply quality cuts
    - reconstruct vertices
  - copy only the final results back to the host (optimised SoA format)
    - convert to legacy format if requested

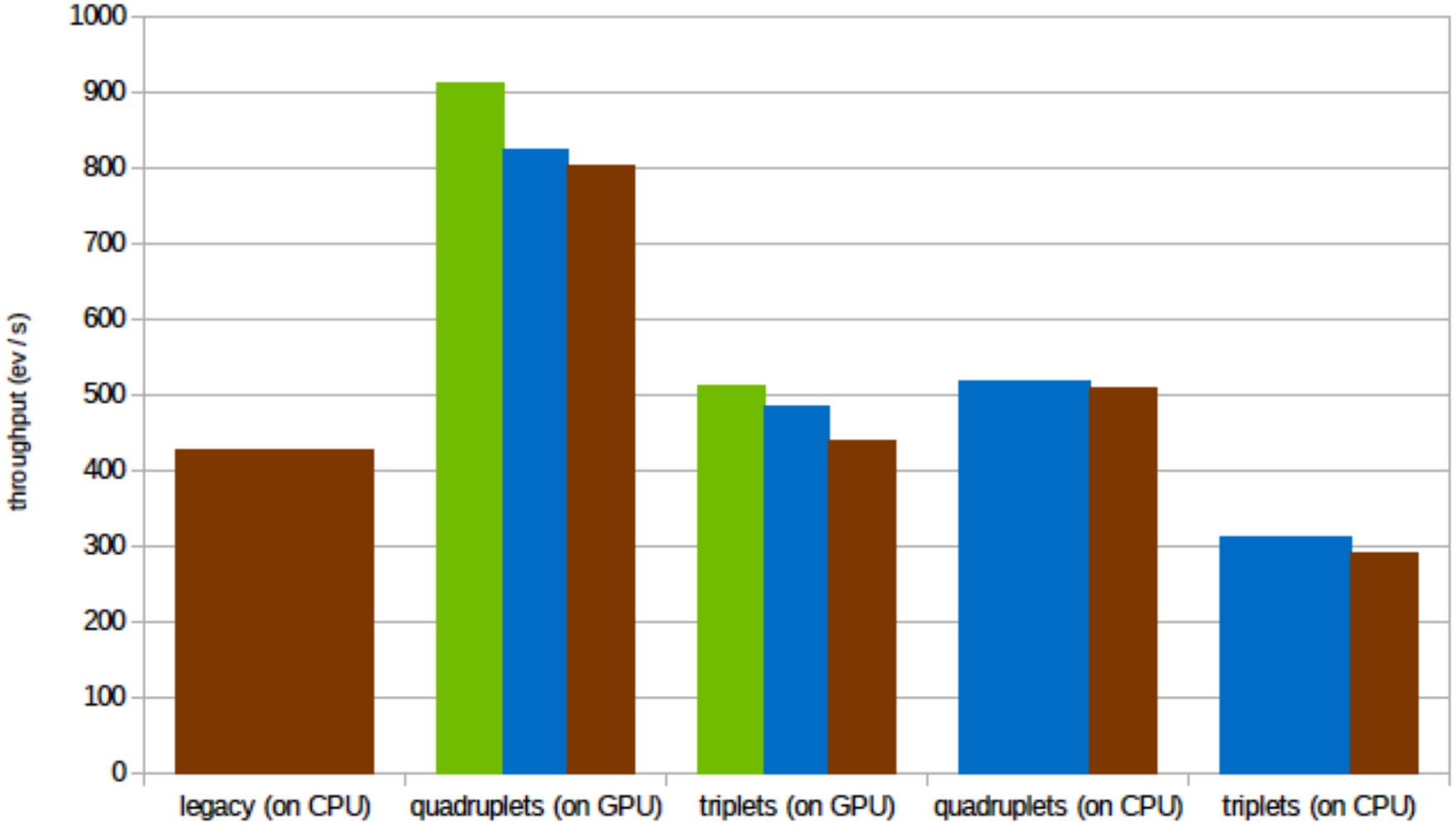


November 7th, 2019 A. Bocci - Heterogeneous online reconstruction at CMS 9

https://indico.cern.ch/event/773049/contributions/3474336

Heterogeneous HLT CMS

improved event throughput



pixel tracks and vertices global reco

CPU

- dual socket Xeon Gold 6130
- 2 × 16 cores (2 x 32 threads)
- throughput measured on a full node
- 4 jobs with 16 threads

GPU

- single NVIDIA Tesla T4
- 2560 CUDA cores
- single job with 10-16 concurrent events

transfer from GPU to CPU

- on demand
- small impact on event throughput

conversion to legacy data formats

- on demand, to be minimised
- small impact on event throughput
- high cost in CPU usage

https://indico.cern.ch/event/773049/contributions/3474336

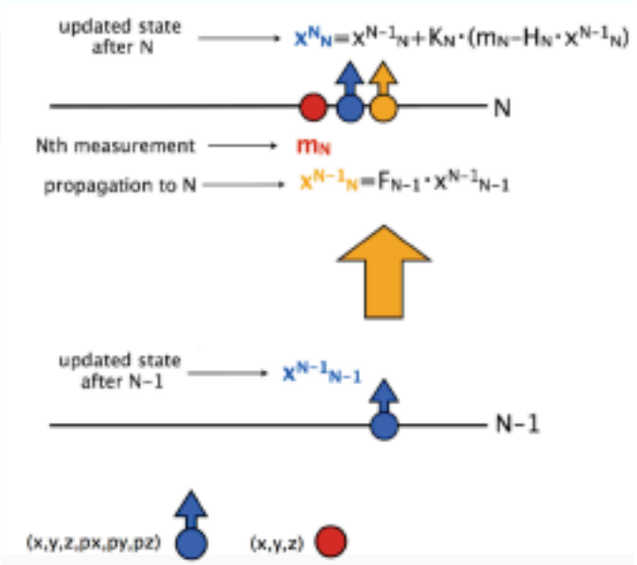
mkFit

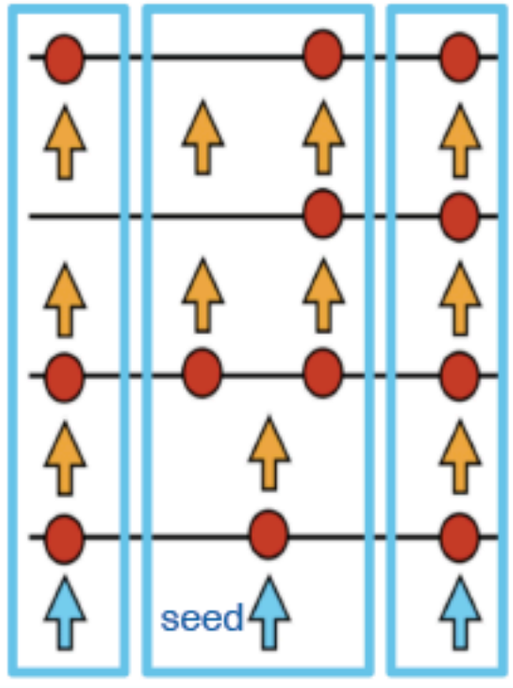
Kalman Filter

- Two step iterative process:
  - **Propagate** the track state from layer N-1 to layer N (prediction)
  - **Update** the state using the detector hit (measurement) on layer N

- Computing **challenges**:
  - Many operations with small matrices, low arithmetic intensity
  - O(2k) seeds and O(100k) hits/event @PU=70

- KF track finding is not straightforward to parallelize
  - Combinatorial algorithm: **branching** to explore many candidates
  - **Heterogeneous** environment:
    different number of hits per track and tracks per event



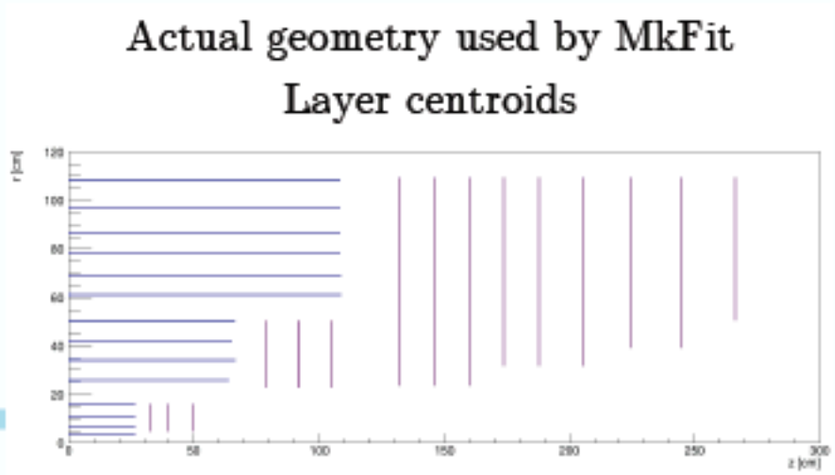


https://indico.cern.ch/event/773049/contributions/3474739

mkFit

Key Features of the Algorithm

- Kalman filter operations use **Matriplex library**: SIMD processing of track candidates
  - auto-generated vectorized code is aware of matrix sparsity
- Algorithm multithreaded at multiple levels with **TBB tasks**
  - events, detector regions, bunches of seeds
- **Lightweight description** of detector in terms of geometry, material, magnetic field
  - collapse barrel (endcap) layers at average r (z), use 3D position of hits
- **Minimize memory operations** (number and size) within combinatorial branching
  - bookkeeping of explored candidates, clone only best ranking ones at each layer (with per seed cap)



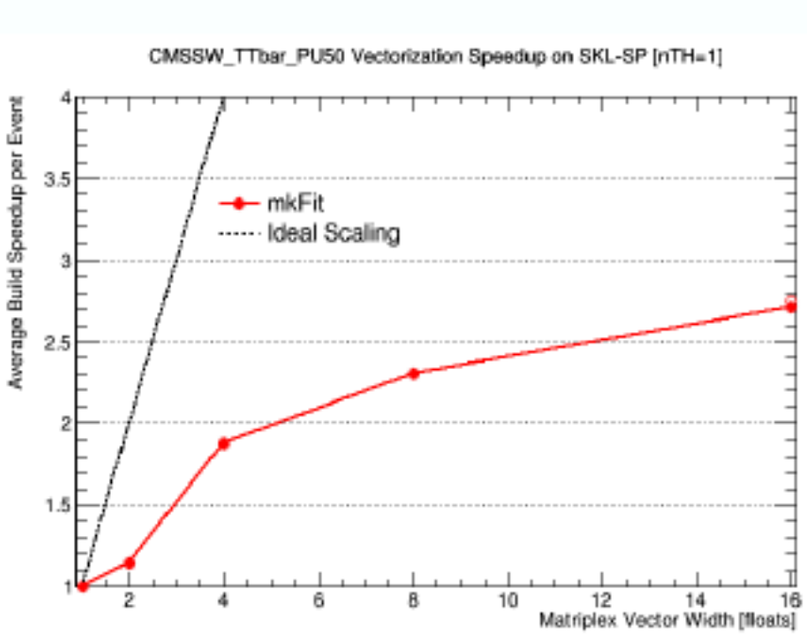
6 2019/11/07 mkFit - CHEP2019

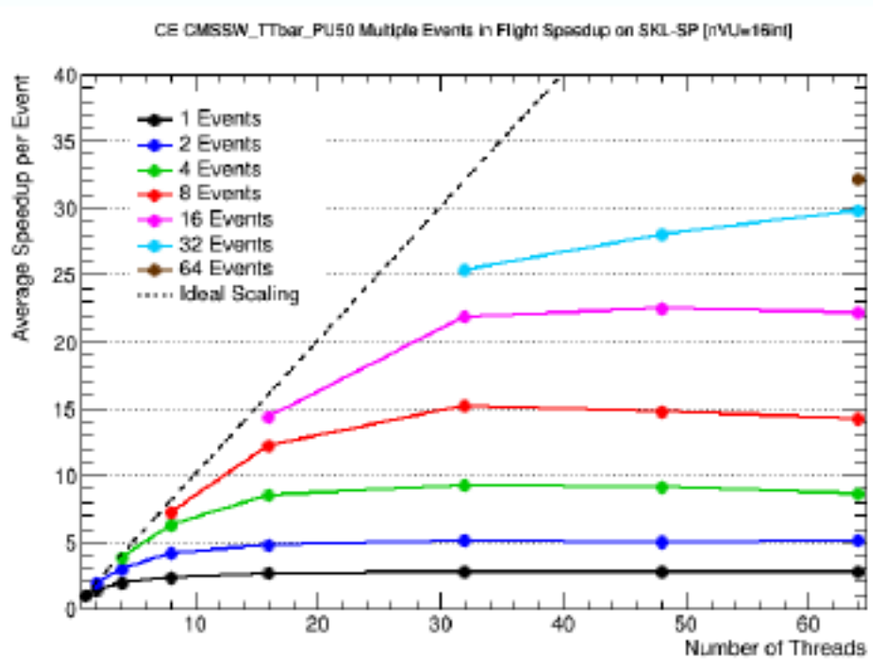
https://indico.cern.ch/event/773049/contributions/3474739

mkFit

Timing Results for Standalone Application

- Showing results on Intel Skylake Gold processor (SKL)
- Core of algorithm achieves nearly **3x** speedup from **vectorization**
  - Ahmdal's law: 60-70% of core algorithm code is effectively vectorized
- Full application achieves **30x** speedup with **multi-threading**
  - close to ideal scaling when all threads dedicated to different events





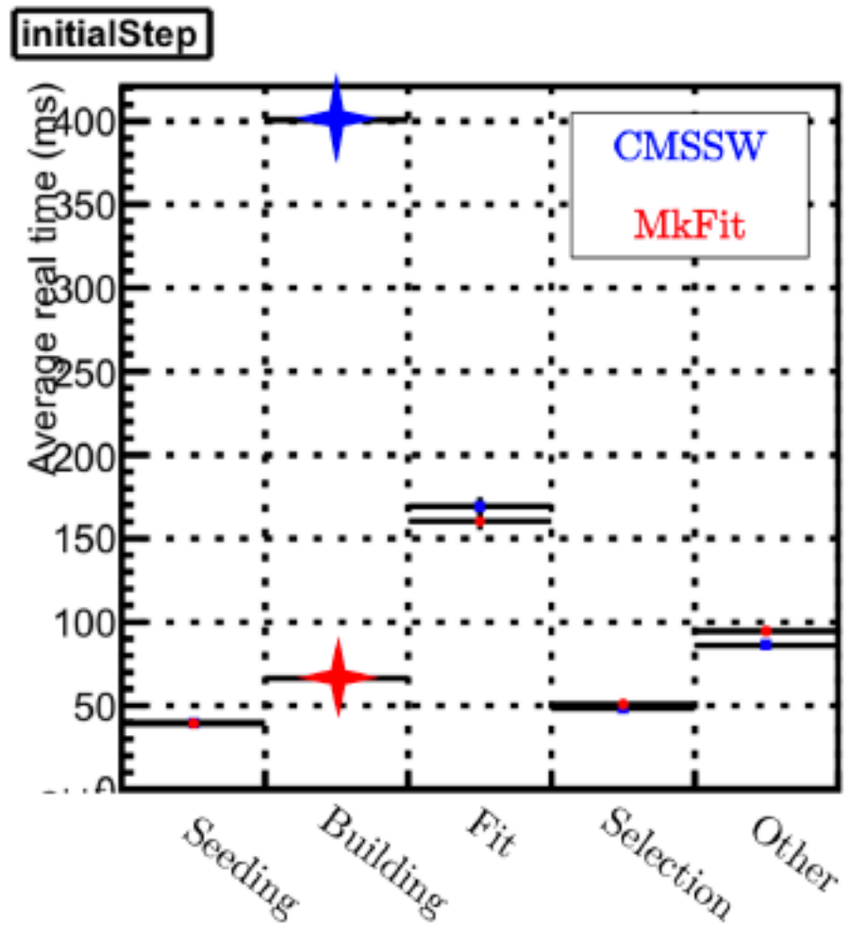
Fermilab

https://indico.cern.ch/event/773049/contributions/3474739

mkFit

Timing Performance of Initial Iteration

- **Single-thread performance** on Intel SKL
  - use ttbar events with <PU>=50
- Speedup of **6.2x** compared to CMSSW
  - track building is not the slowest component anymore!
- Data format **conversions** between CMSSW and mkFit account for **~25%** of mkFit time
  - larger speedup possible if data formats are harmonized
- Here mkFit is compiled with icc and AVX-512
  - with gcc speedup reduces to ~2.5x



Fermilab

10 2019/11/07 mkFit - CHEP2019

https://indico.cern.ch/event/773049/contributions/3474739

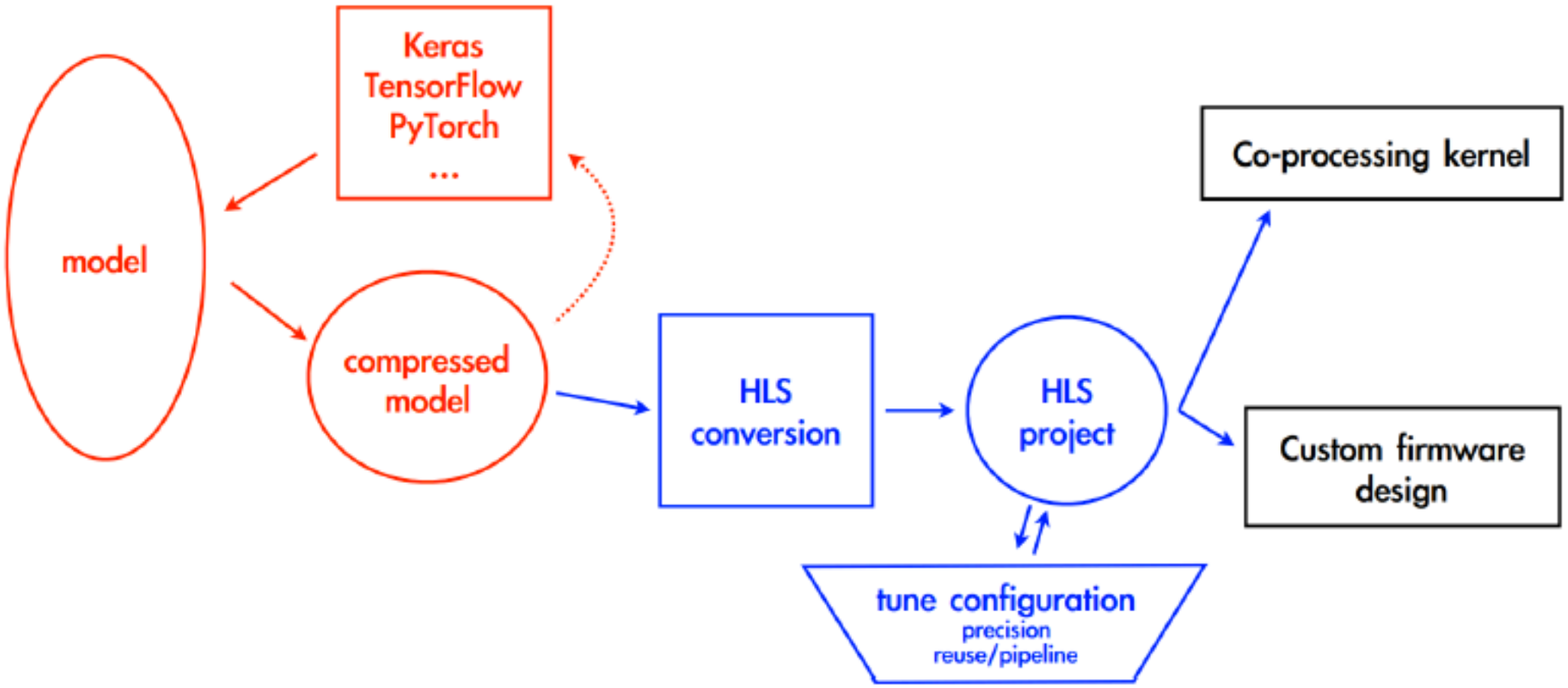
ML on FPGAs

- ***hls4ml*** is a software package for creating HLS implementations of neural networks
  - https://hls-fpga-machine-learning.github.io/hls4ml/
- Supports common layer architectures and model software
- Highly customizable output for different latency and size needs
- Simple workflow to allow quick translation to HLS

https://indico.cern.ch/event/797510/contributions/3313676

ML on FPGAs

Design Workflow



- Design model with standard software tools (Keras, Tensorflow, PyTorch)
- Pass network architecture and weights/biases along with configuration parameters to `hls4ml` (creates HLS project)
- Interface HLS code with desired project

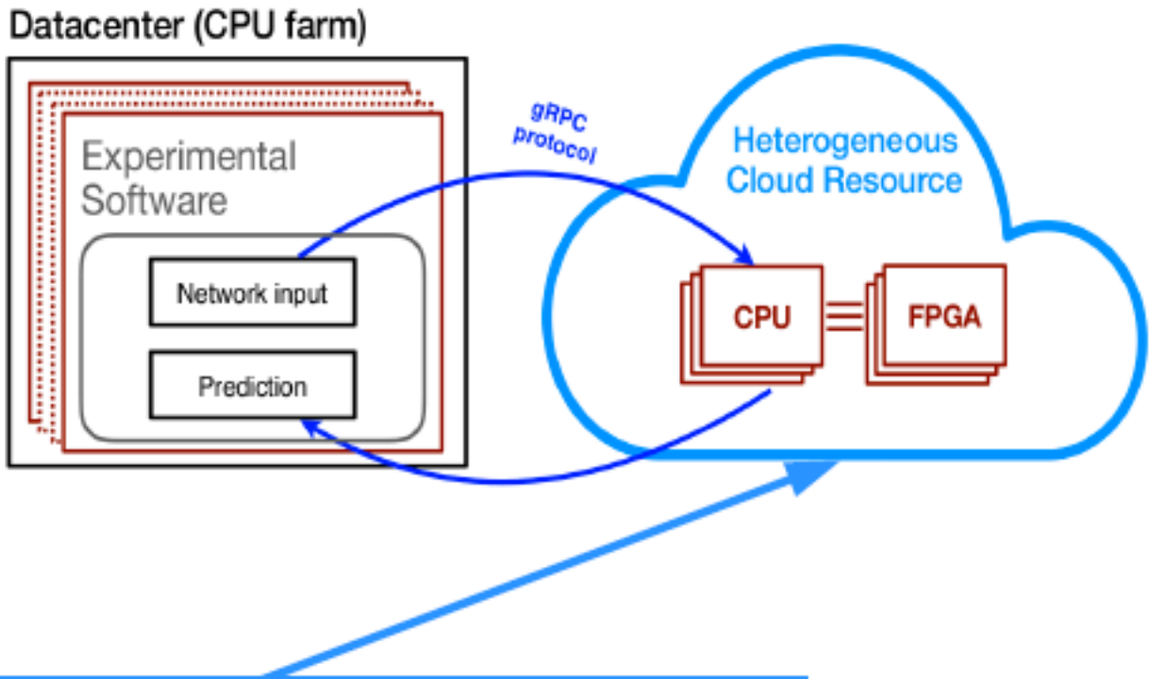
17

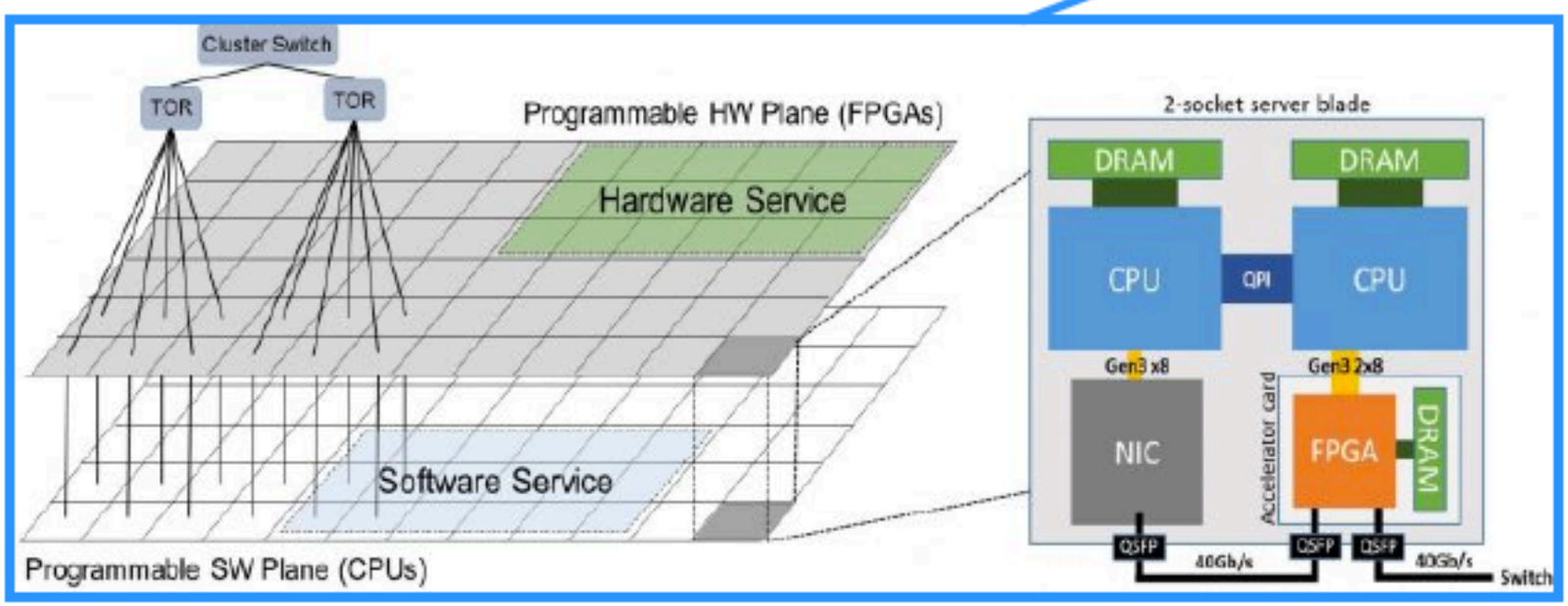
https://indico.cern.ch/event/797510/contributions/3313676

ML on FPGAs

Microsoft Brainwave

- Microsoft Brainwave is a CPU-FPGA server farm
- ML offered “as a service”
  - Send a preprocessed image to Brainwave, get prediction back from ResNet50
- Extremely interesting option for longer latency (>10 ms) systems
- Stay tuned for another talk in the future





36

https://indico.cern.ch/event/797510/contributions/3313676

FPGA inference as a service

SONIC in CMSSW

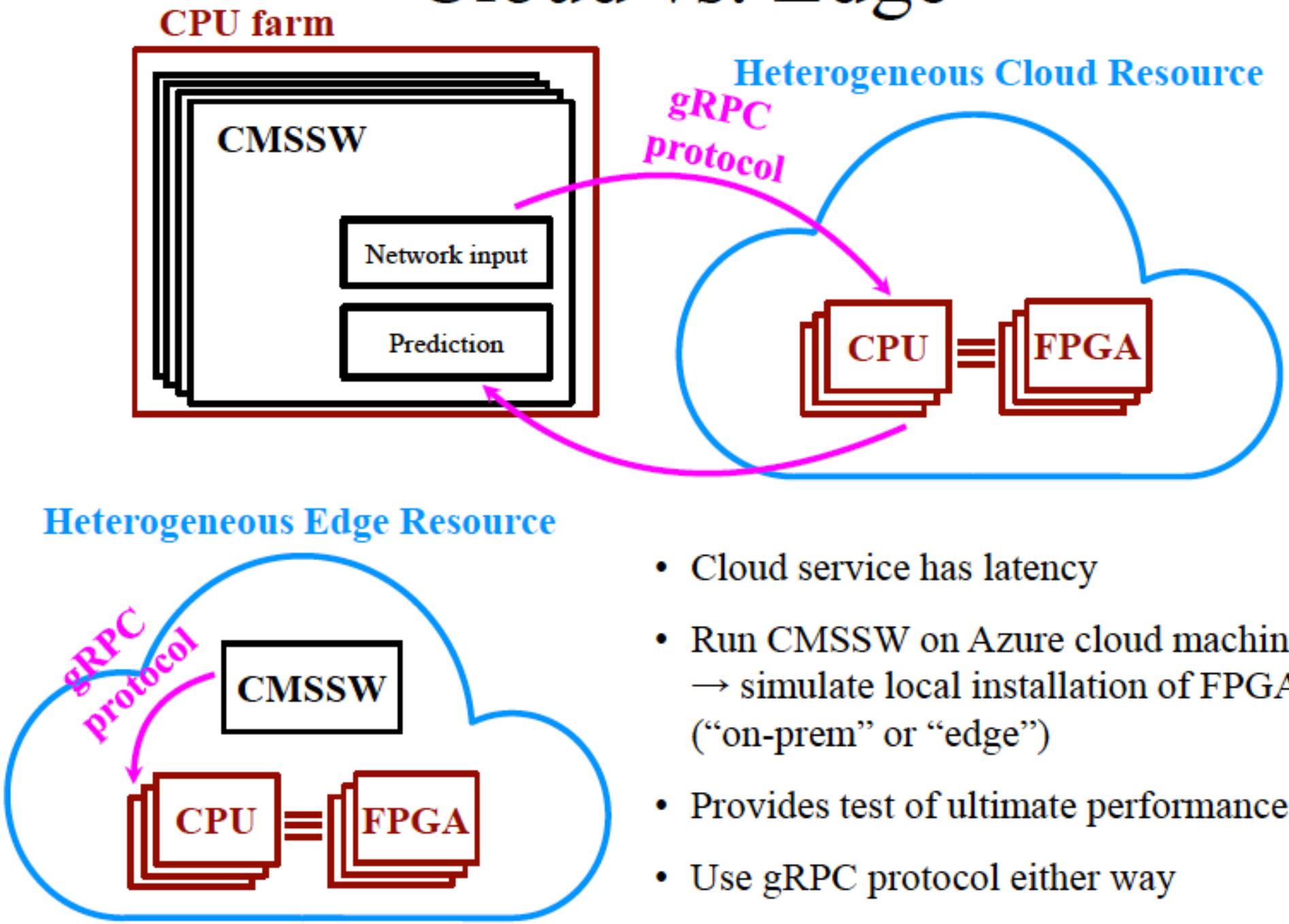
- **S**ervices for **O**ptimized **N**etwork **I**nference on **C**oprocessors
  - Convert experimental data into neural network input
  - Send neural network input to coprocessor using communication protocol
  - Use ExternalWork mechanism for asynchronous requests
- Currently supports:
  - gRPC communication protocol
    - Callback interface for C++ API in development
      → wait for return in lightweight std::thread
  - TensorFlow w/ inputs sent as TensorProto (protobuf)
- Tested w/ Microsoft Brainwave service (cloud FPGAs)
- gRPC SonicCMS repository on GitHub

HOW2019 Kevin Pedro 12

https://indico.cern.ch/event/759388/contributions/3303405 https://arxiv.org/abs/1904.08986

FPGA inference as a service

Cloud vs. Edge



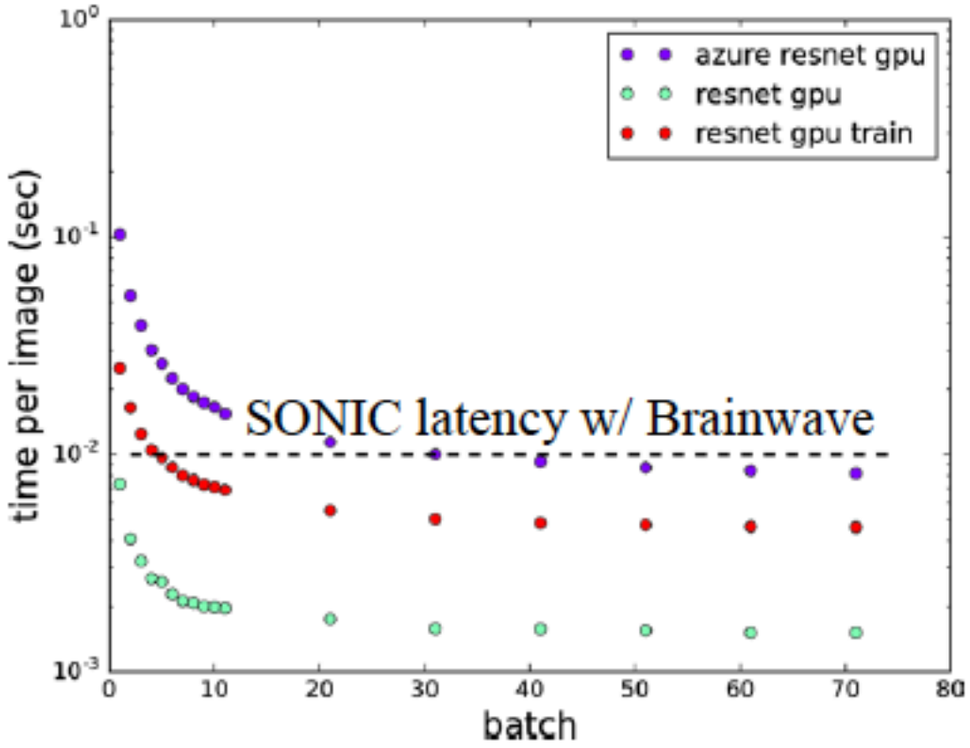
- Cloud service has latency
- Run CMSSW on Azure cloud machine → simulate local installation of FPGAs ("on-prem" or "edge")
- Provides test of ultimate performance
- Use gRPC protocol either way

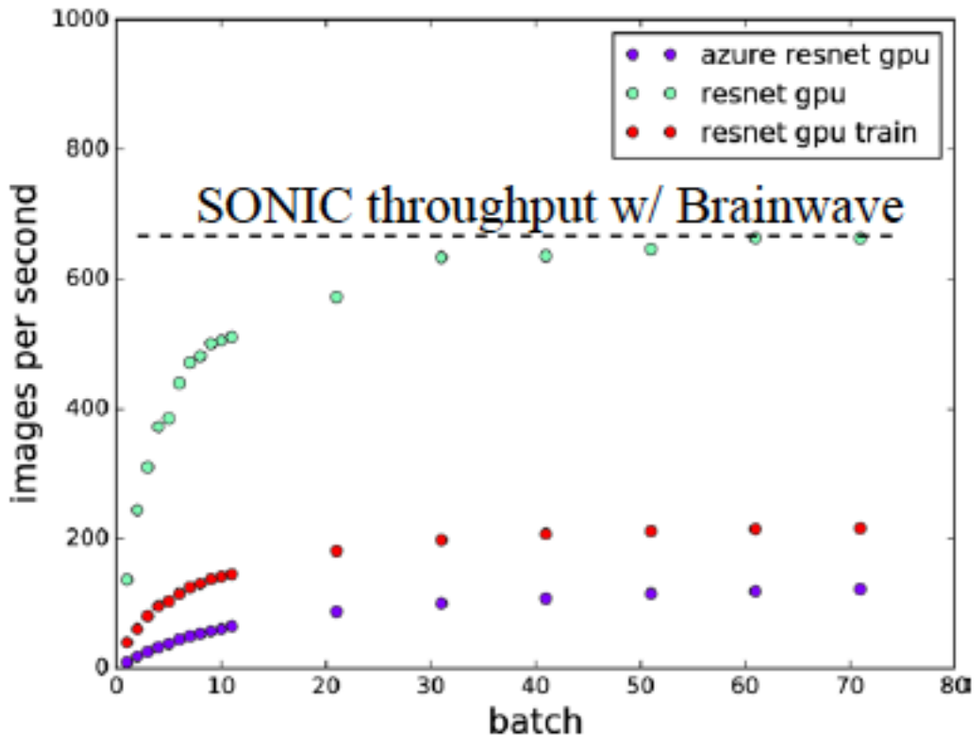
HOW2019 Kevin Pedro 13

https://indico.cern.ch/event/759388/contributions/3303405 https://arxiv.org/abs/1904.08986

FPGA inference as a service

GPU Performance





- Above plots use NVidia GTX 1080, TensorFlow v1.10
- GPU directly connected to CPU via PCIe
- TF built-in version of ResNet-50 performs better on GPU than quantized version used in Brainwave

HOW2019 Kevin Pedro 18

https://indico.cern.ch/event/759388/contributions/3303405 https://arxiv.org/abs/1904.08986

Summary

- **Reconstruction and software trigger R&D is reach with opportunities in the next decade**
- **The overreaching goal is to maximize success of the leading experimental HEP facilities, dominated by the interests of the HL-LHC experiments**
- **HEP Software Foundation (HSF) identified (7) activities of leading interest for the 2020s in**
- **IRIS-HEP Innovative Algorithms focus area is supporting a range of R&D activities fitting the roadmap of HEP computing in the 2020s**

- **A large fraction of topical studies presented in major HEP computing conferences on reconstruction and software trigger topics go towards solving the challenges of the next decade**
  - ✓ A somewhat selective digest of CHEP 2019 presentations is available in the backup

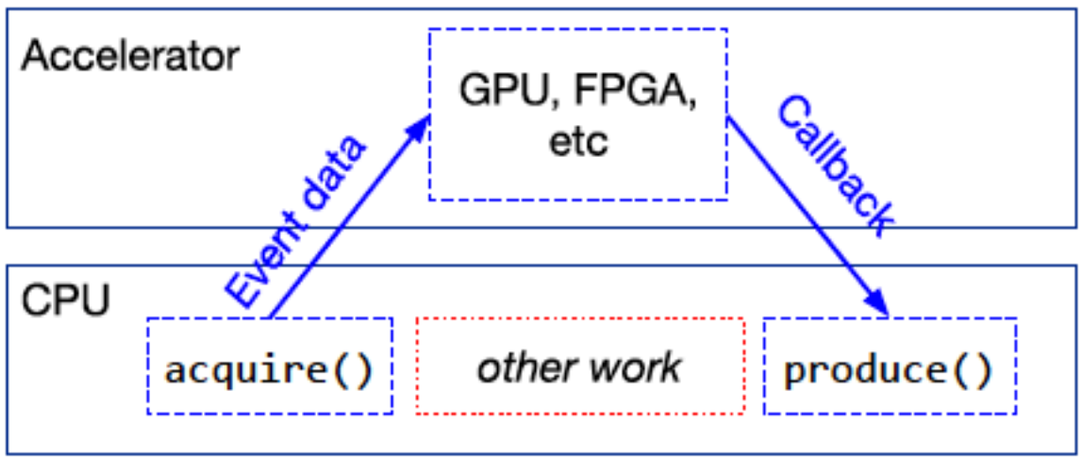
BACKUP

CMS heterogeneous framework

Fermilab

External worker concept

- Replace blocking waits with a callback-style solution
- Traditionally the algorithms have one function called by the framework, `produce()`
- That function is split into two stages
  - `acquire()`: Called first, launches the asynchronous work
  - `produce()`: Called after the asynchronous work has finished
- `acquire()` is given a reference-Counted smart pointer to the task that calls `produce()`
  - Decrease reference count when asynchronous work has finished
  - Capable of delivering exceptions



CMS

https://indico.cern.ch/event/773049/contributions/3473266

CMS heterogeneous framework

Fermilab

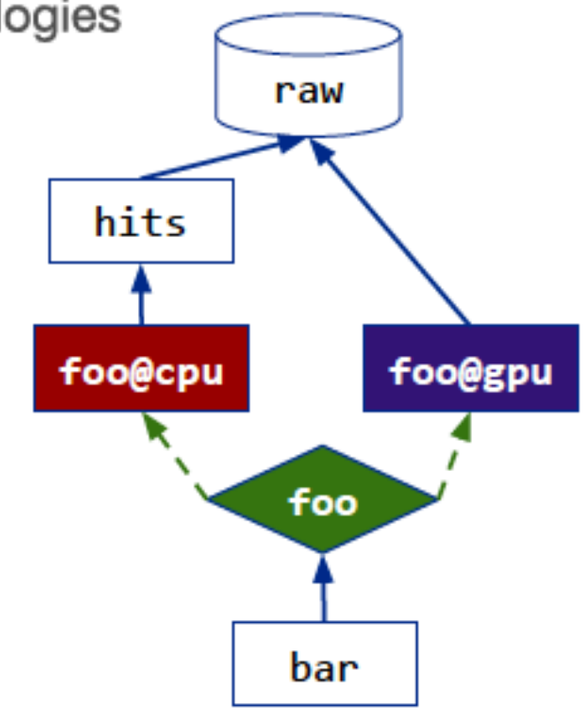
Switch mechanism for producers

- SwitchProducer added to configuration
  - Allows specifying multiple modules associated to same module label
  - At runtime picks one to be run based on available technologies
    - Consumers dictate which producers are run

```
hits = Producer("HitsProducer",
    input = "raw"
)

foo = SwitchProducer(
    cpu = Producer("FooProducer",
        input = "hits"),
    gpu = Producer("FooProducerGPU",
        input = "raw")
)

bar = Producer("BarProducer",
    input = "foo"
)
```



CMS

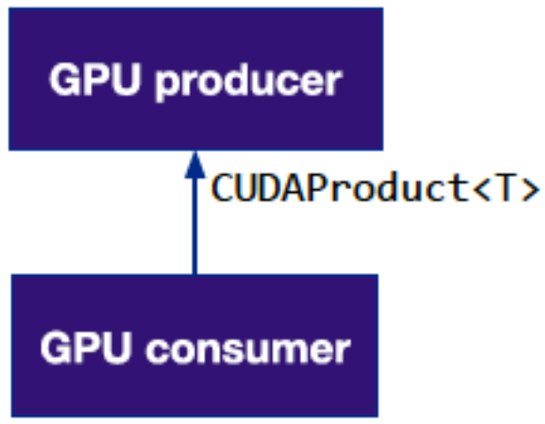
https://indico.cern.ch/event/773049/contributions/3473266

CMS heterogeneous framework

Fermilab

CUDA pattern: sharing resources between modules

- We introduced a wrapper template `CUDAProduct<T>` for a product of type `T`
  - Product `T` is partly or fully in GPU memory
- Wrapper holds the devide id and CUDA stream used to produce the product
  - Also CUDA event to mark the completion of asynchronous processing in case that was not finished when the module ended
- Consumer module uses
  - The same device
  - Either the same CUDA stream, or another that synchronizes with the input CUDA stream
- Two types of modules
  - Normal: launch work without synchronization
  - External worker: if need to transfer anything back to CPU and synchronize



CMS

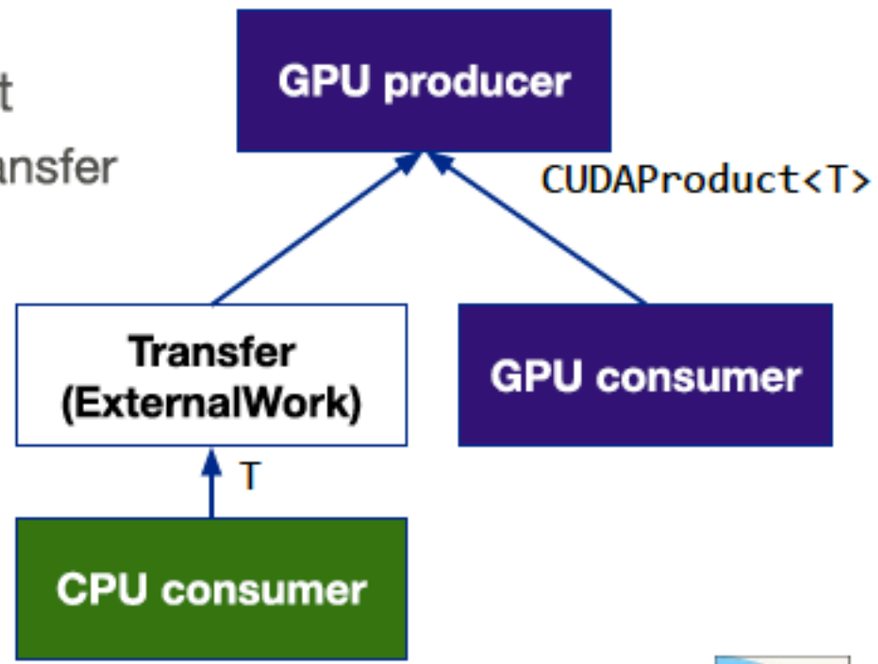
https://indico.cern.ch/event/773049/contributions/3473266

CMS heterogeneous framework

Fermilab

CUDA pattern: minimizing data movements

- Add additional modules to do the transfers
- Output product type is different anyway between CPU and GPU
  - At minimum `T` vs. `CUDAProduct<T>`
- Exploit framework's behavior to run a producer only if some other module consumes the product
  - I.e. if no-one asks for the product in CPU, do not transfer



CMS

https://indico.cern.ch/event/773049/contributions/3473266

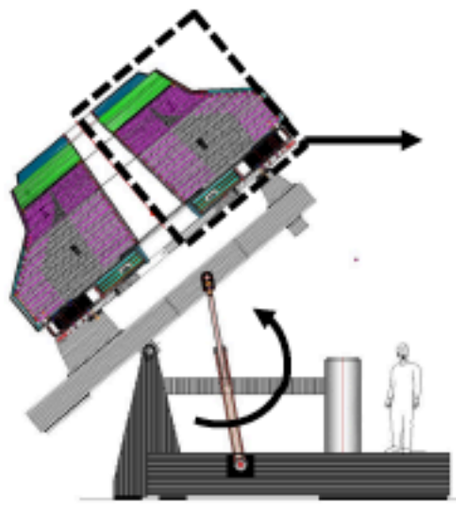
HGCAL clustering with GPU

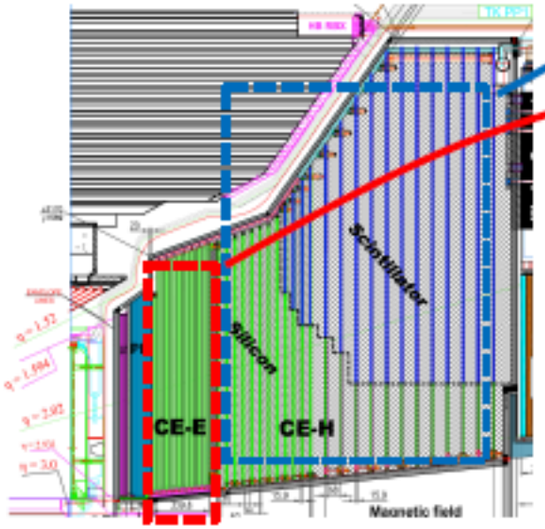
CMS High Granularity Calorimeter

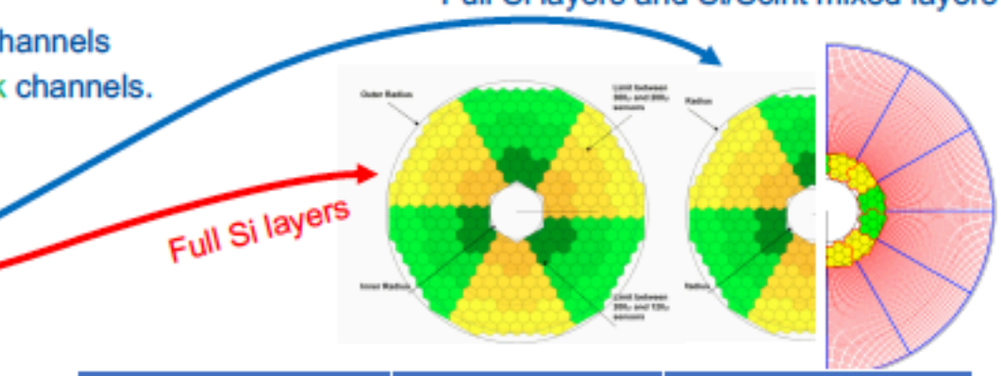
- Current CMS endcap ECAL and HCAL calorimeters will be replaced by High Granularity Calorimeter (HGCAL), a sampling calorimeter system based on Si sensors and plastic scintillators, during CMS phase-II upgrade.

- Plots from "The Phase-2 Upgrade of the CMS endcap calorimeter Technical Design Report" http://cds.cern.ch/record/2293646/files/

- Full system operates at −35℃ maintained by a CO_2 cooling system
- Covers $1.5 < |\eta| < 3.0$ on both left and right sides
- Total size z=2m, r = 2.3m. Total weight 215 ton per endcap
- 620 m² of Silicon sensors (120/200/300 um). 0.5-1.0 cm² → 6M channels
- 400 m² of plastic scintillators with SiPM readout. 4-30 cm² → 240k channels.





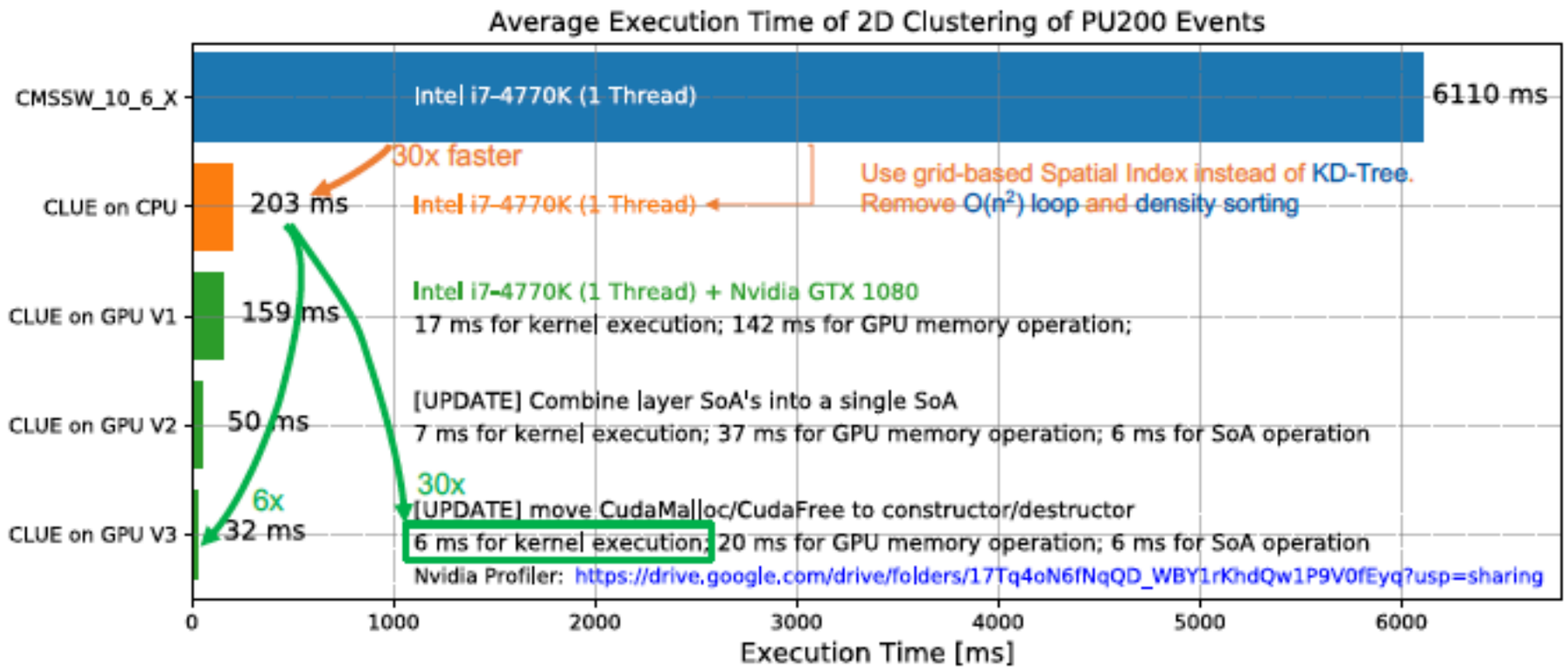


	CE-E	CE-H
Number of layers	28 layer/endcap	8+14 layer/endcap
Sensor type	Si	Si, Scintillator
Absorber type	Cu, CuW, Pb	Stainless Steel, Cu
Thickness	$25X_0$, 1.3λ	~9.5λ

https://indico.cern.ch/event/773049/contributions/3473275

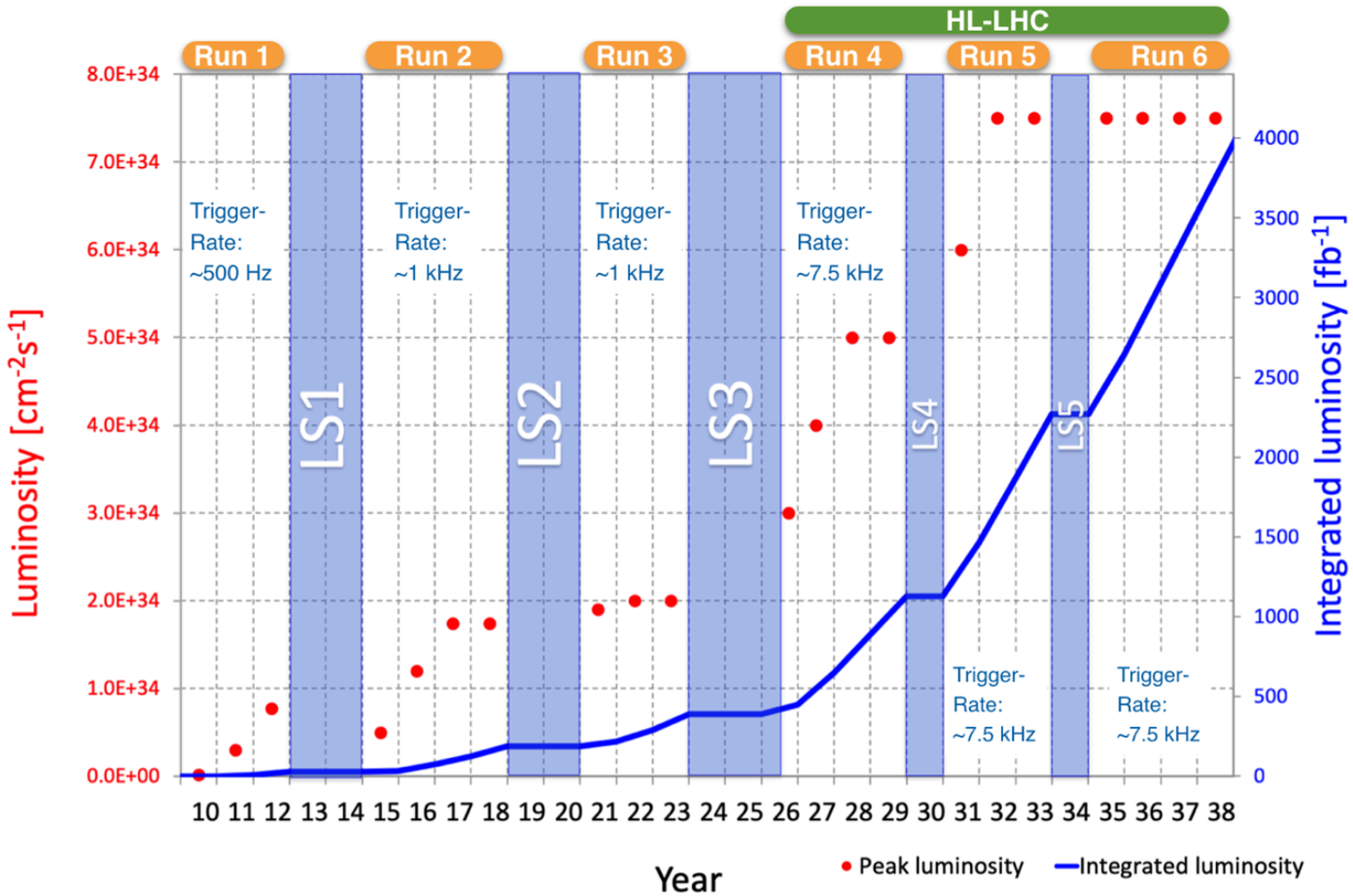
HGCAL clustering with GPU

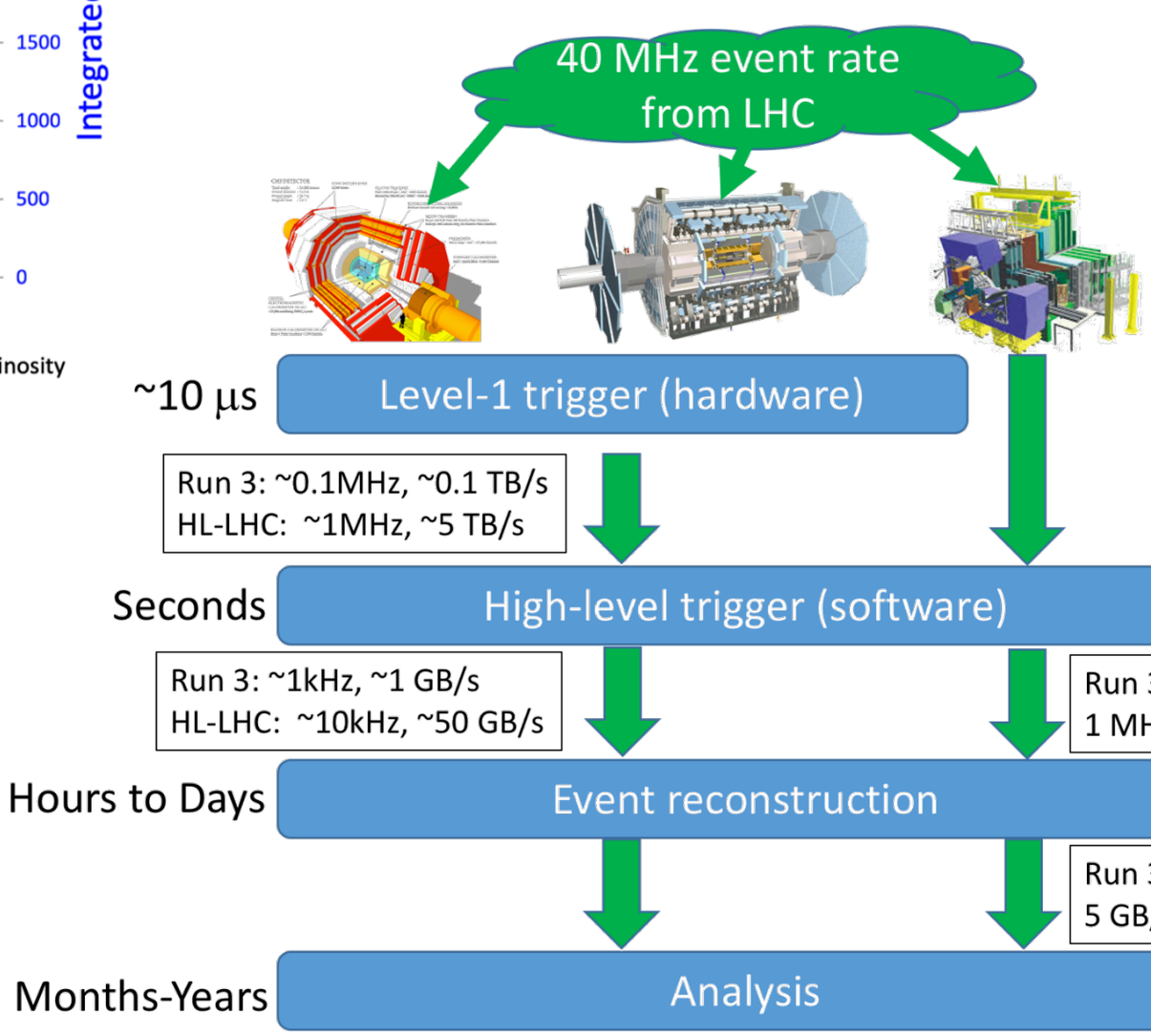
CLUE in CMS Software (CMSSW) Framework



- CLUE CPU has about 30X speed up over previous clustering in CMSSW.
- CLUE GPU V3 gives an additional 6X on top of CLUE CPU in CMSSW framework.
- GPU run time (32 ms) also includes memcpy between host and device (20 ms) and SoA conversion (6 ms). But they can be shared by other GPU reconstruction steps and can be partially hidden if multiple CUDA stream work on different events. 30X speed up over CPU if excluding memcpy and SoA conversion.

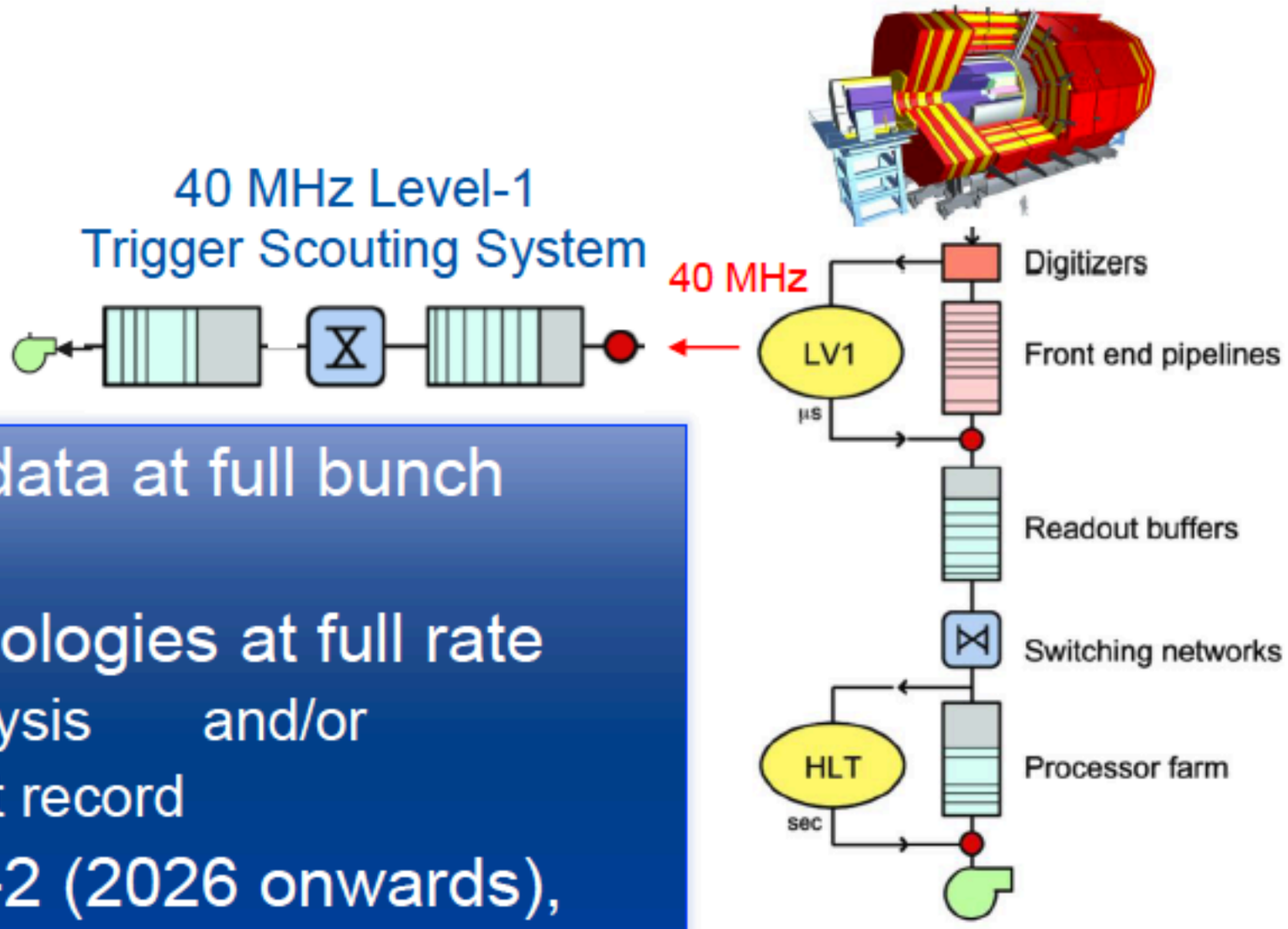
https://indico.cern.ch/event/773049/contributions/3473275


HL-LHC
Run 1
Run 2
Run 3
Run 4
Run 5
Run 6
LS1
LS2
LS3
LS4
LS5
Trigger-Rate: ~500 Hz
Trigger-Rate: ~1 kHz
Trigger-Rate: ~1 kHz
Trigger-Rate: ~7.5 kHz
Trigger-Rate: ~7.5 kHz
Trigger-Rate: ~7.5 kHz
Luminosity [cm-2s-1]
Integrated luminosity [fb-1]
8.0E+34
7.0E+34
6.0E+34
5.0E+34
4.0E+34
3.0E+34
2.0E+34
1.0E+34
0.0E+00
4000
3500
3000
2500
2000
1500
1000
500
0
10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38
Year
Peak luminosity
Integrated luminosity


40 MHz event rate from LHC
~10 μs
Level-1 trigger (hardware)
Run 3: ~0.1MHz, ~0.1 TB/s
HL-LHC: ~1MHz, ~5 TB/s
Seconds
High-level trigger (software)
Run 3: ~1kHz, ~1 GB/s
HL-LHC: ~10kHz, ~50 GB/s
Run 3: 1 MHz
Hours to Days
Event reconstruction
Run 3: 5 GB/s
Months-Years
Analysis

CMS towards continuous analysis

L1 Trigger Scouting



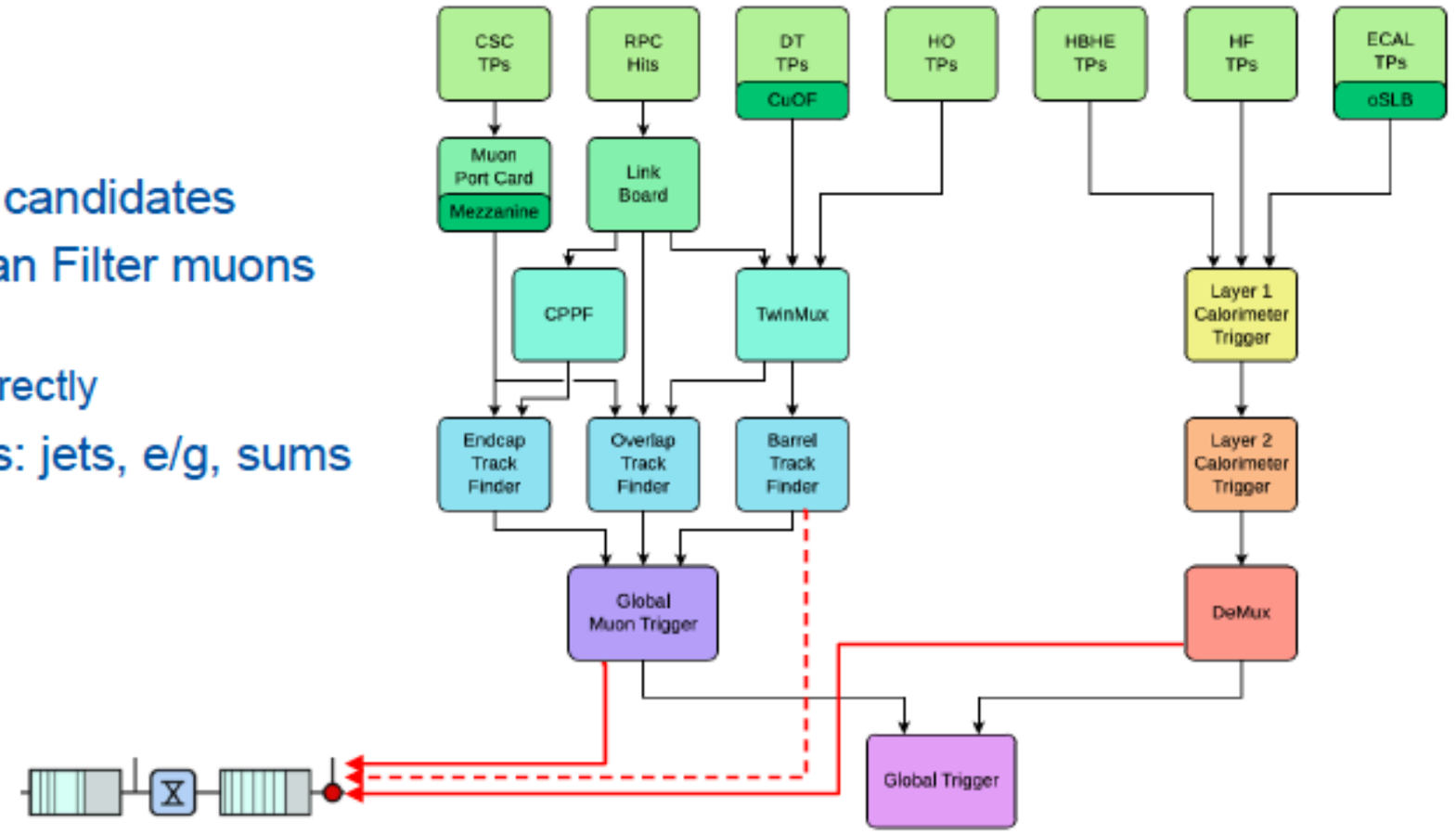
- Acquire L1 trigger data at full bunch crossing rate
- Analyze certain topologies at full rate
  - semi real-time analysis and/or
  - storing of tiny event record
- Planned for Phase-2 (2026 onwards), Prototyping now

https://indico.cern.ch/event/773049/contributions/3474327

CMS towards continuous analysis

Plan for Run-3 (2021): Muon + Calo Scouting

- When: starting 2021
- Capture @ 40 MHz
  - Up to 8 final muon candidates
  - Barrel Muon Kalman Filter muons (displaced Muons)
    - Through GMT or directly
  - Calorimeter objects: jets, e/g, sums



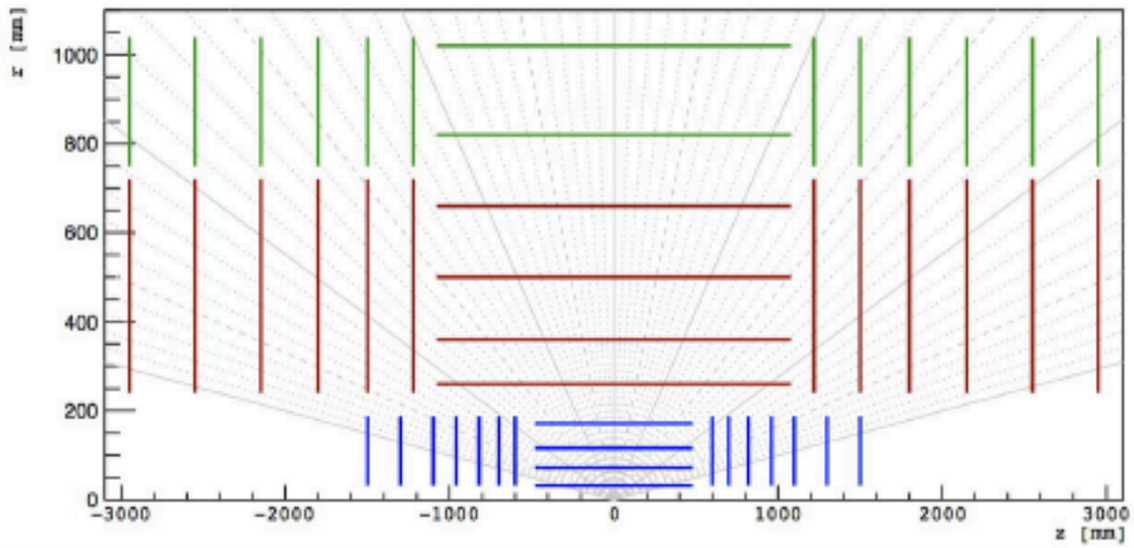
40 MHz Scouting Prototype System

https://indico.cern.ch/event/773049/contributions/3474327

Track ML

Challenge Datasets

- Accurate simulation engine (ACTS) to produce realistic dataset
  - One file with list 3D points
  - Ground truth : one file with point to particle association
  - Ground truth auxiliary : true particle parameter (origin, direction, curvature)
  - Typical events with ~200 parasitic collisions (~10.000 tracks/event, ~100k hits/event)
- The goal of the challenge is to **assemble hits into tracks**
- Large training sample 100k events, 10 billion tracks ~100 GB

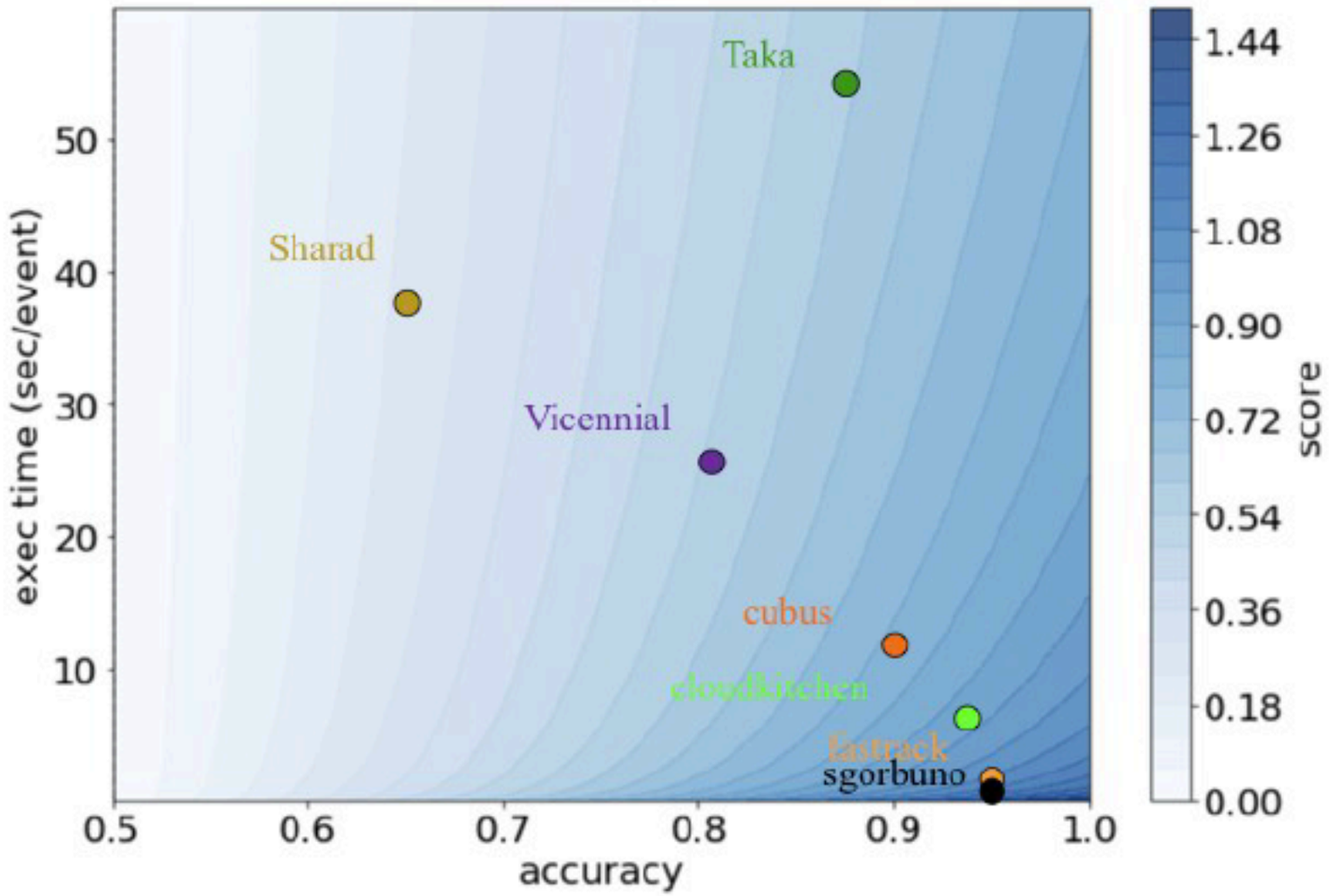


11/05/19 TrackML Challenge, CHEP19, J.-R. Vlimant 6

https://indico.cern.ch/event/773049/contributions/3474831

Track ML

Time – Accuracy Decomposition



Incidentally, best solutions are also best accuracy and best timing.
Software will be submitted and analyzed in the coming weeks.

https://indico.cern.ch/event/773049/contributions/3474831

Track ML

Insights on Algorithms

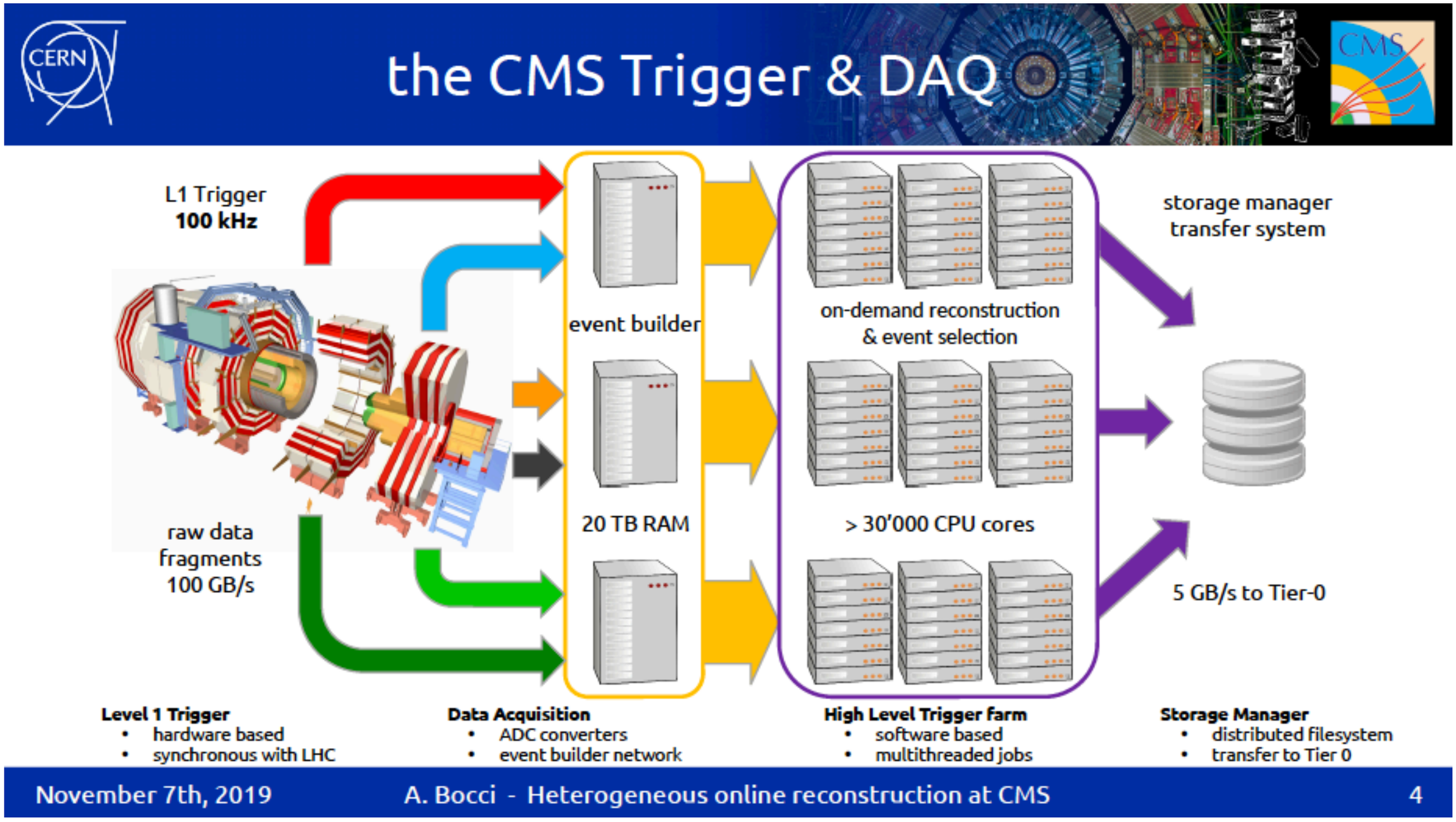
- **First : *Sgorbuno***
  - Sergey Gorbunov is a physicist, expert in tracking
  - 3rd position at the accuracy phase
  - Iterative steps, triplet seeding, **trajectory following**

- **Second** : ***fastrack***
  - Dmitry Emeliyanov is a physicist
  - Graph representation of neighbors, cellular automaton, track following

- **Third : *cloudkitchen***
  - Marcel Kunze is a former physicist,
  - Direct acyclic graph of voxels, pair and triplet classification + Top Quark solution of accuracy phase (**trajectory following**, track cleaning, all with **machine learning** for quality selection)

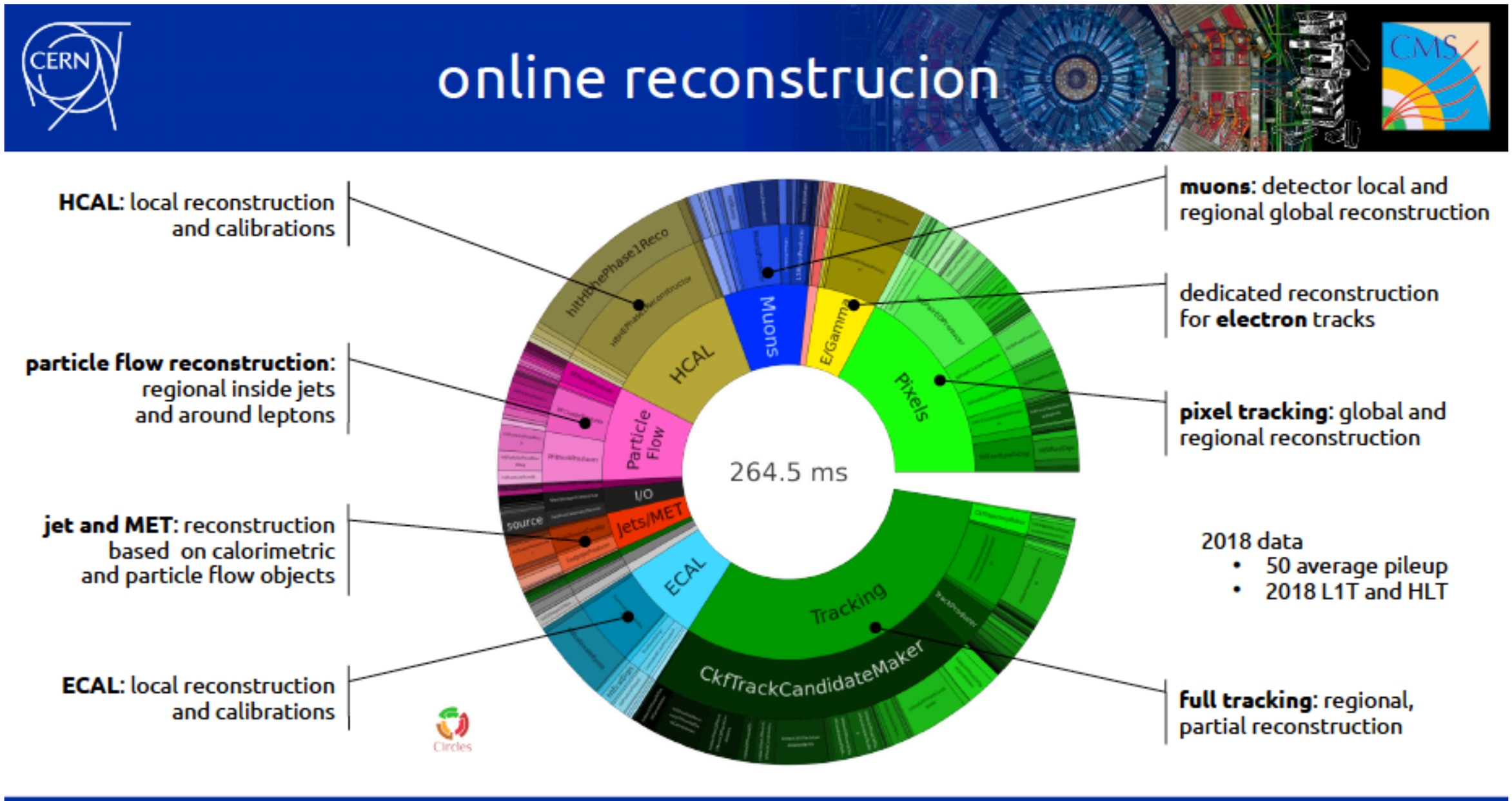
https://indico.cern.ch/event/773049/contributions/3474831

Heterogeneous HLT CMS



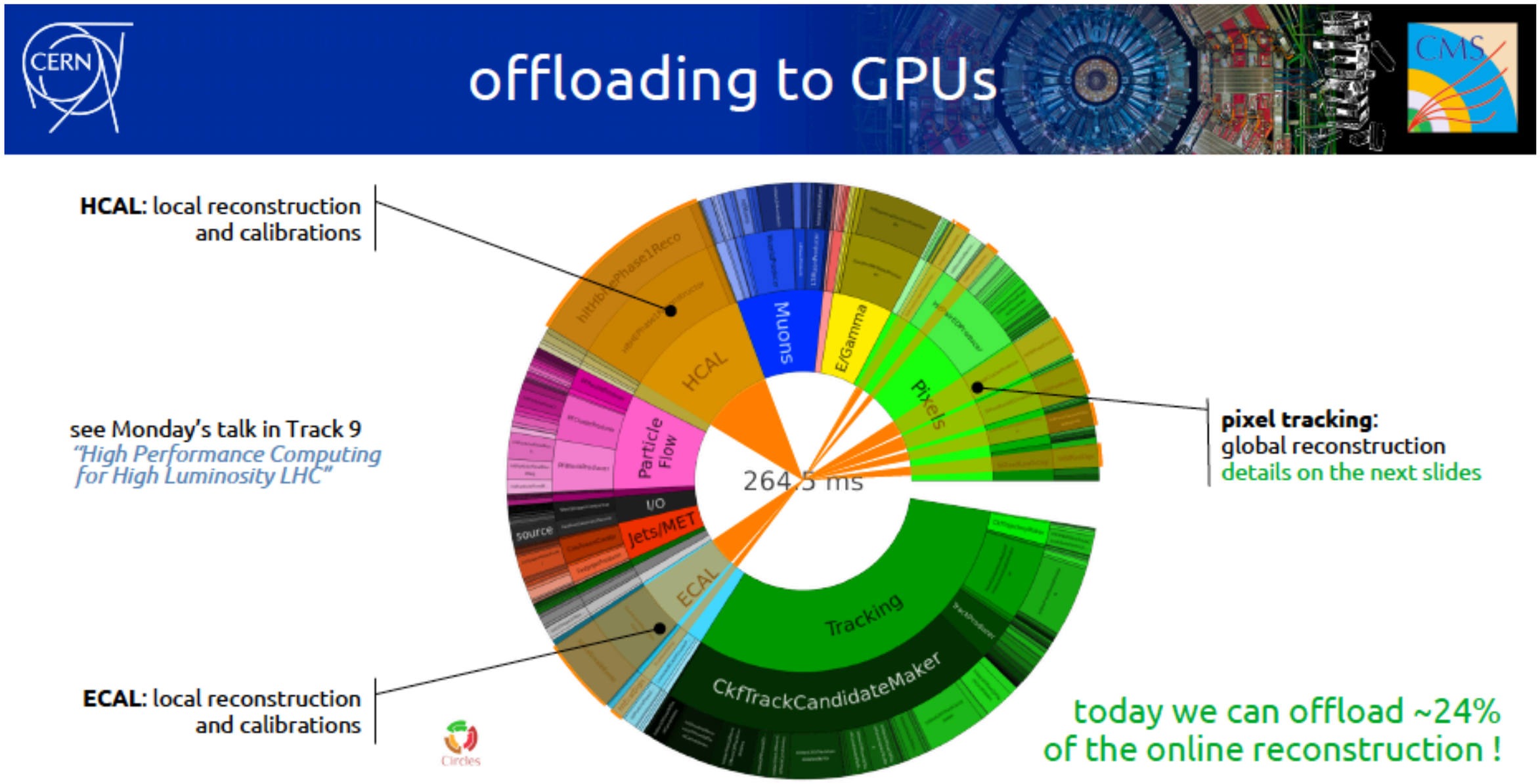
https://indico.cern.ch/event/773049/contributions/3474336

Heterogeneous HLT CMS



https://indico.cern.ch/event/773049/contributions/3474336

Heterogeneous HLT CMS



https://indico.cern.ch/event/773049/contributions/3474336

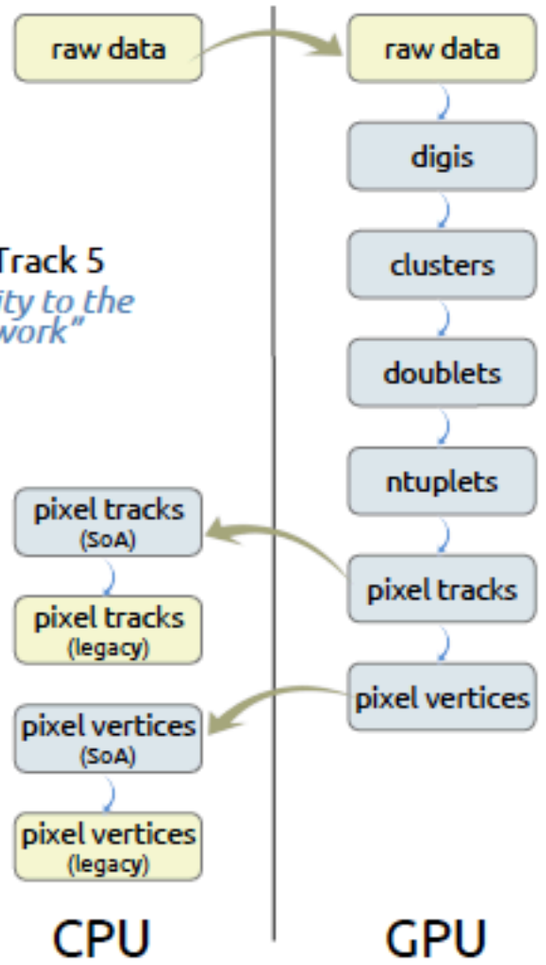
Heterogeneous HLT CMS

the Patatrack demonstrator

- the overall approach
  - reconstruct pixel-based tracks and vertices on the GPU
  - leverage existing support for threads and on-demand reconstruction
  - minimise data transfer

see Monday's talk in Track 5
"Bringing heterogeneity to the CMS software framework"

- the full workflow
  - copy the raw data to the GPU
  - run multiple kernels to perform the various steps
    - decode the raw data
    - cluster the pixel hits
    - form hit doublets
    - form hit ntuplets (triplets or quadruplets) with a Cellular automaton algorithm
    - clean up duplicates
  - take advantage of the GPU computing power to improve the physics
    - fit the track parameters (Riemann fit, broken line fit) and apply quality cuts
    - reconstruct vertices
  - copy only the final results back to the host (optimised SoA format)
    - convert to legacy format if requested

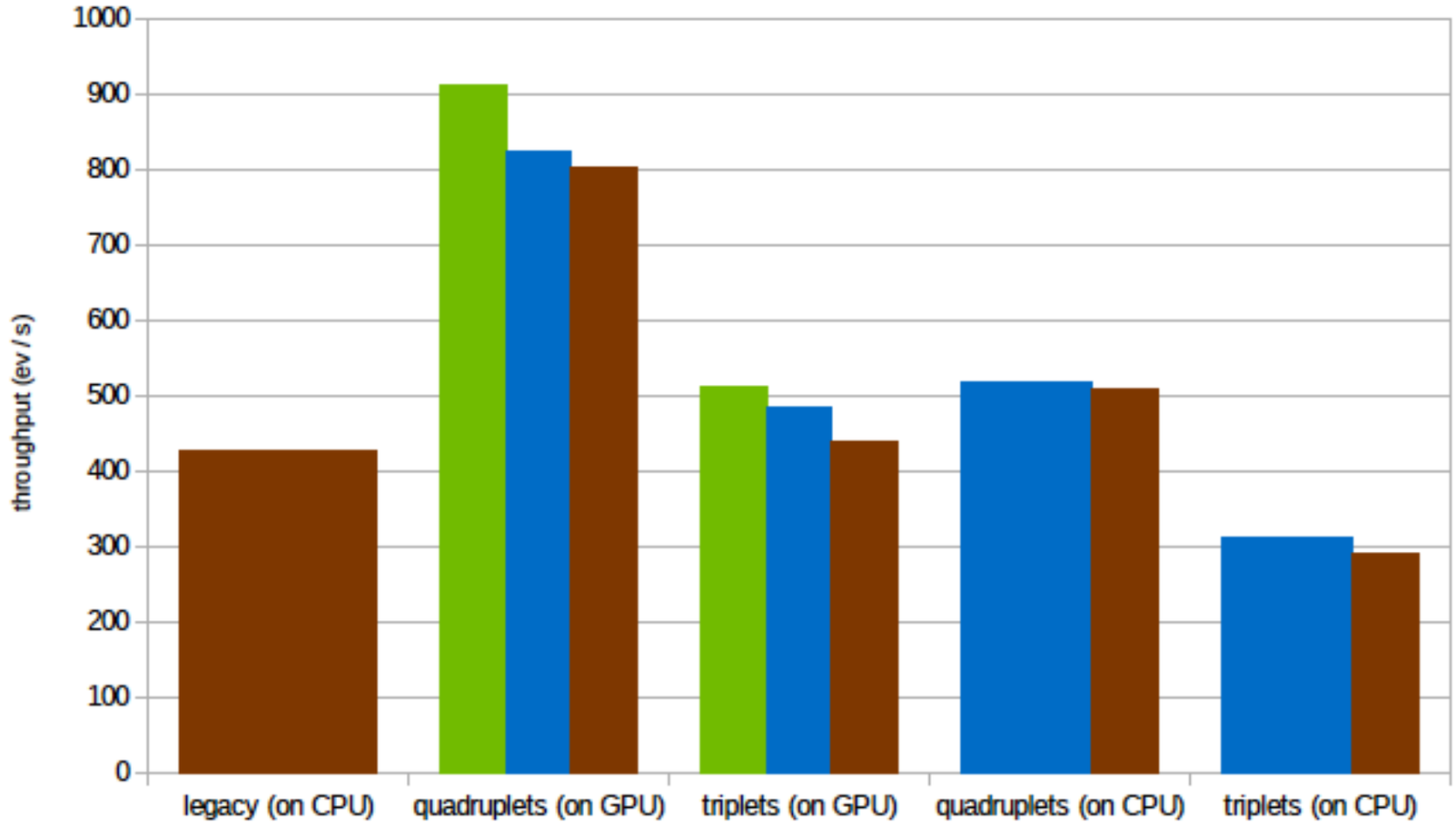


November 7th, 2019 A. Bocci - Heterogeneous online reconstruction at CMS 9

https://indico.cern.ch/event/773049/contributions/3474336

Heterogeneous HLT CMS

improved event throughput



pixel tracks and vertices global reco

CPU

- dual socket Xeon Gold 6130
- 2 × 16 cores (2 x 32 threads)
- throughput measured on a full node
- 4 jobs with 16 threads

GPU

- single NVIDIA Tesla T4
- 2560 CUDA cores
- single job with 10-16 concurrent events

transfer from GPU to CPU

- on demand
- small impact on event throughput

conversion to legacy data formats

- on demand, to be minimised
- small impact on event throughput
- high cost in CPU usage

https://indico.cern.ch/event/773049/contributions/3474336

Heterogeneous HLT towards Phase-2

we have a problem !

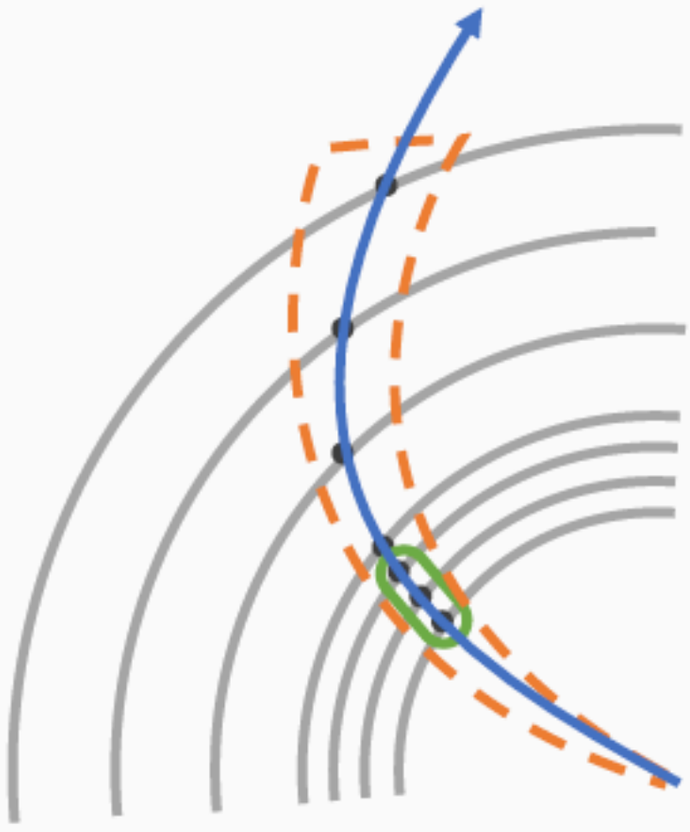
- higher pileup
  - optimistic extrapolation: × 3
- new detectors
  - based on MC: × 1.3
- higher L1 trigger rate
  - design: **× 7.5**
- "Phase 2" HLT resource needs
  - optimistically … × 30
- expected improvements in CPUs
  - optimistic: × 4
  - realistic: × 2

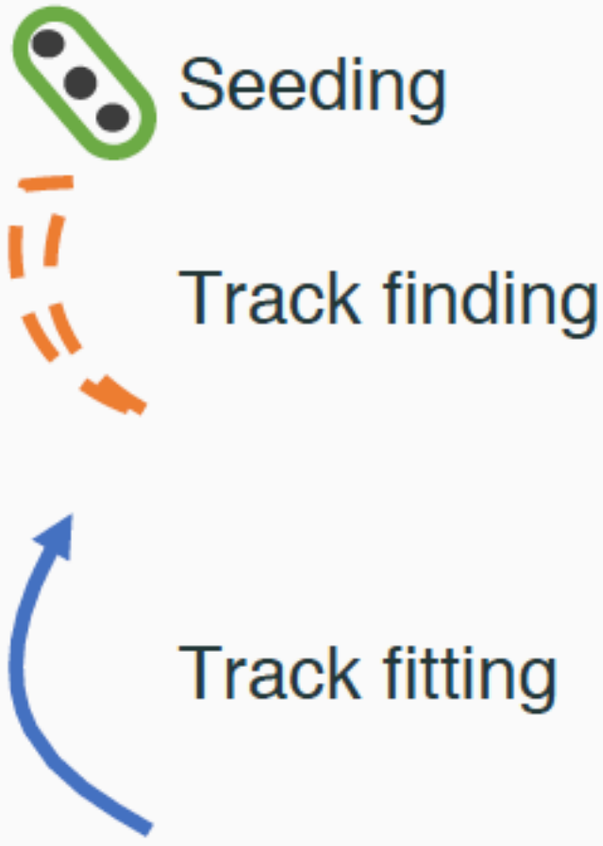
March 13th, 2019 ACAT 2019 - Towards a heterogeneous computing farm for the CMS HLT 14

https://indico.cern.ch/event/708041/contributions/3276337

A Common Tracking Software (ACTS)

Track reconstruction





Paul Gessinger 31.07.2019 - IRIS-HEP topical meeting 2

https://indico.cern.ch/event/830160/contributions/3476465

The Acts project

- Based on the ATLAS tracking software[2]
- Got started to enable R&D in faster cycles, test thread-safety concepts

acts-core / ACTS-267

Make ACTS const-correct (again)

1. Drop all occurences of "mutable" in ACTS (~~ACTS-256~~)
2. Get rid of static variables and class members (~~ACTS-309~~)
3. Replace pointer-to-mutable by pointer-to-const where appropriate (ACTS-208)
4. Drop incorrect usage of pointer-to-const where the target is modified (ACTS-269)
5. Clean up ugly interfaces discovered in ACTS by introducing const-correctness (ACTS-270), and in particular rework geometry building (ACTS-276)

- Contributions come mostly from ATLAS people, but there is increasing involvement from CEPC, Belle-2, FCC-hh and more

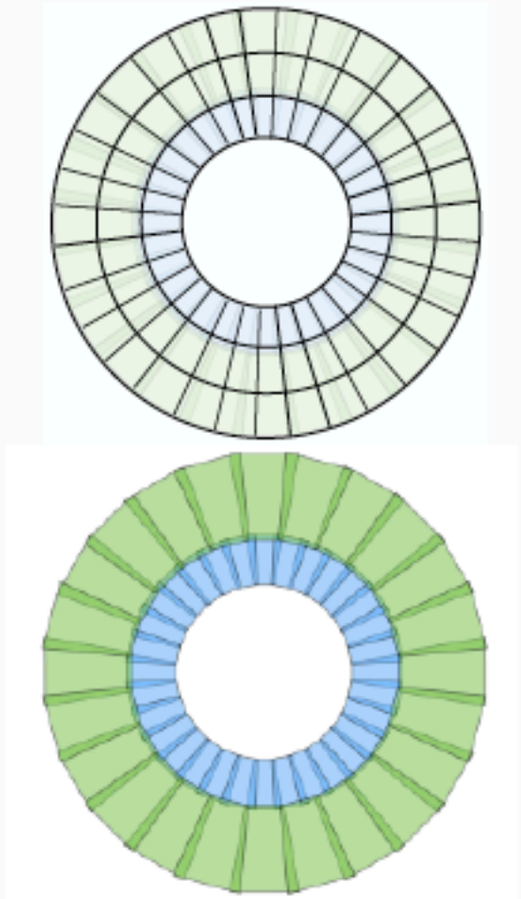
Basic design goals

- Provide a toolbox with experiment-**independent** components
- Should allow building (possibly) experiment-**specific** *applications* like seeding, track finding, vertex finding
- Minimal dependencies (e.g. **no** required ROOT dependency)
- Minimize **dynamic memory** allocation
- Be **thread-safe** out of the box (thread local state)
- Limit (ideally eliminate) virtual inheritance: **concepts**!
- Rigorously **tested**, use **continuous integration**

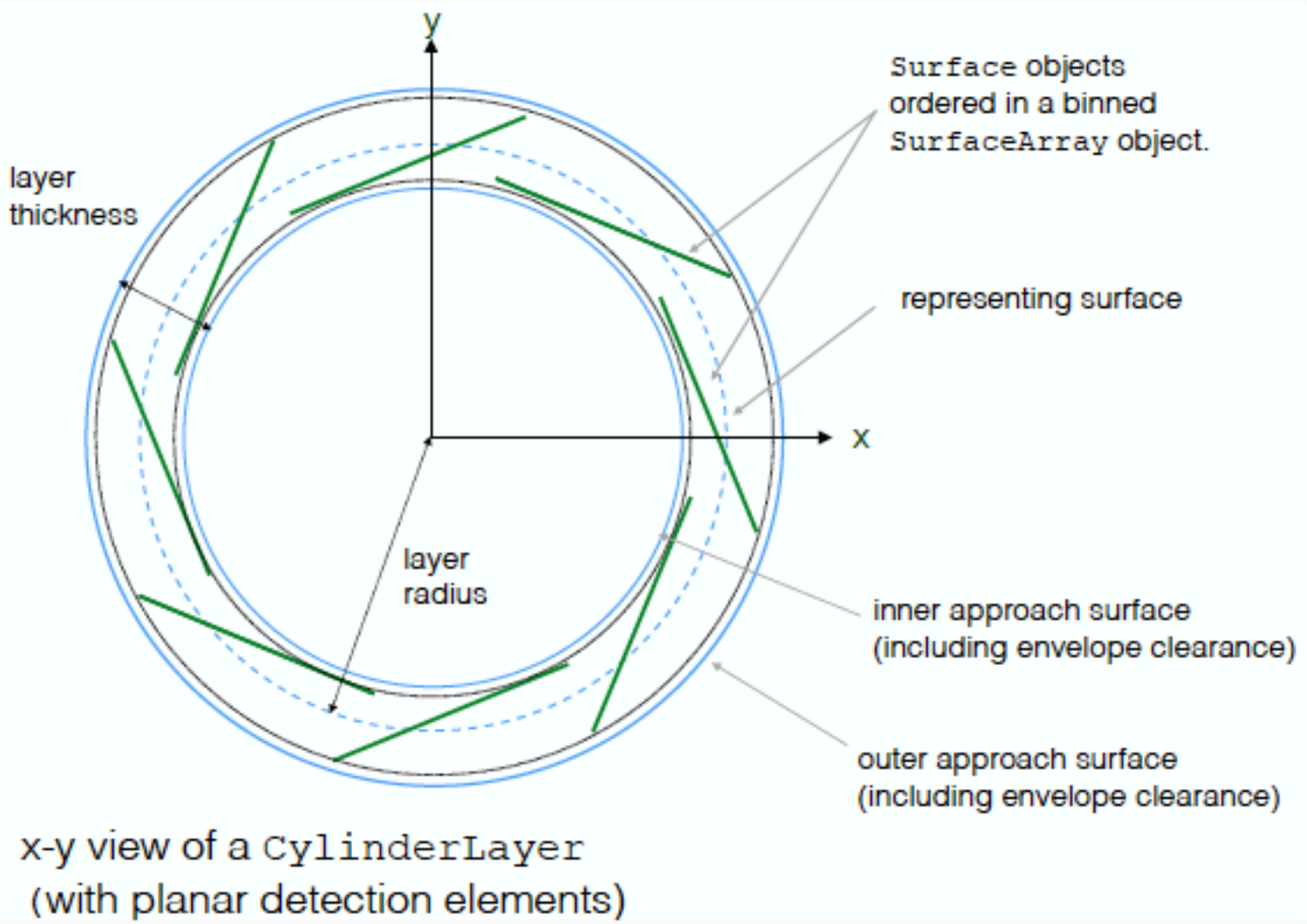
Project structure

- `acts-core`: main library
  - Contains tools and components
  - Doesn't assume anything about event-processing framework
- `acts-framework`: small `GaudiHive`-inspired event processing framework
  - Event-level parallelism for testing
  - Has *generic* geometry, `TGeo` and `DD4hep` plugins
- `acts-fatras`: `Acts`-based fast track simulation
  - Can be used to create scenarios for testing and validation
- Licensed under **MPLv2**

Geometry modeling and navigation



A. Salzburger



x-y view of a `CylinderLayer`
(with planar detection elements)

Propagation in Acts

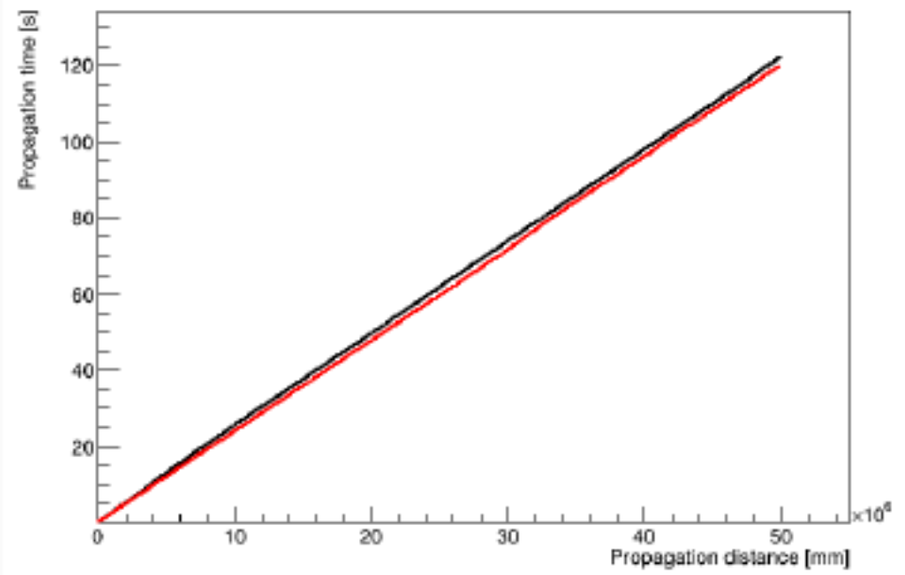
- Generalize *optional* components into **Actors** and **Aborters** (e.g. `MaterialActor`)
- Keep component structure for *required* components: **Stepper** and **Navigator**
- All components are template parameters: **no virtual calls**
- ATLAS' different propagators → *integration-term extensions* to our main integrator **EigenStepper**

Timing information

- Timing is integrated into numerical integration
- Data structures allow for time measurements naturally (see KF later)
- Tests indicate no negative performance impact

$$\vec{x} = (x, y, z, \mathbf{t}, T_x, T_y, T_z, q/p)$$

with time, without time



MR by F. Klimpel

Status

- Propagation, single Kalman Filter, seed finder developed and usable
- Time propagation implemented and tested
- So far: tests mostly standalone
- Tests of ATLAS geometry (ID + first tests for calorimeter modeling and navigation)
- Demonstrated multi-threaded execution with alignment

Plans

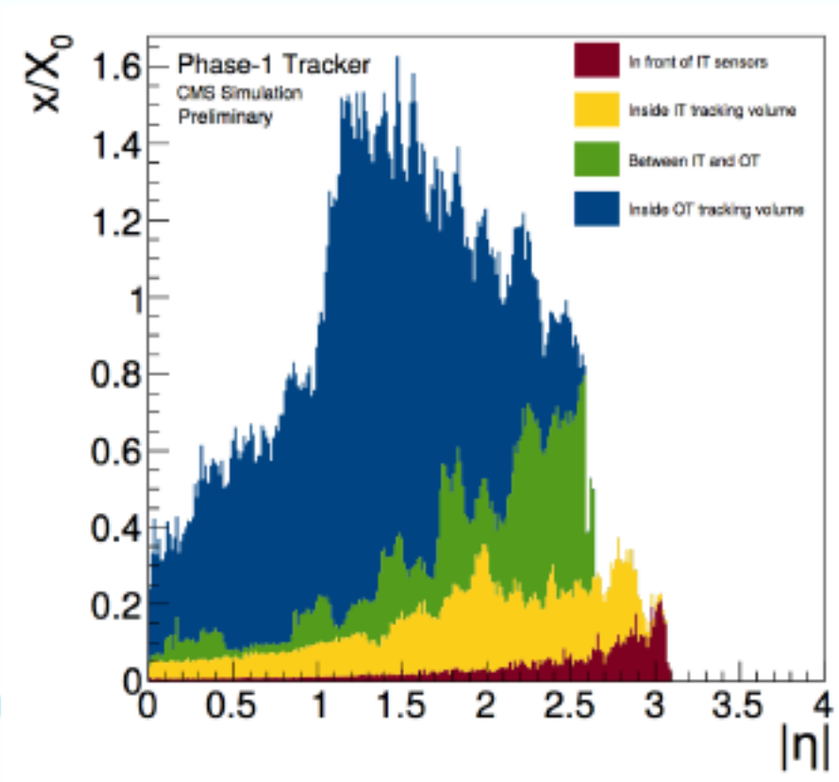
- Develop better test integration in ATLAS (i.e. same interface but Acts under the hood)
- Compare Acts KF to ATLAS KF
- Rewrite navigation to make it more adaptable to various navigation approaches
- Start work on a Combinatorial Kalman Filter implementation (some preliminary tests exists)
- Get to the point where we can demonstrate a full track reconstruction chain

mkFit

mkFit Project

- Ongoing for ~5 years, well advanced
- Collaboration between physicists and computer scientists from Cornell, Fermilab, Princeton, UCSD, UOregon
  - funding from NSF IRIS-HEP, DOE SciDAC, USCMS
  - http://trackreco.github.io/

- Mission: **speedup Kalman filter (KF) tracking** algorithms using highly parallel architectures
- Why sticking to KF?
  - Widely used in HEP in general, and CMS in particular
  - Demonstrated **high efficiency** physics performance
  - Robust handling of **material effects**



4 2019/11/07 mkFit - CHEP2019

https://indico.cern.ch/event/773049/contributions/3474739

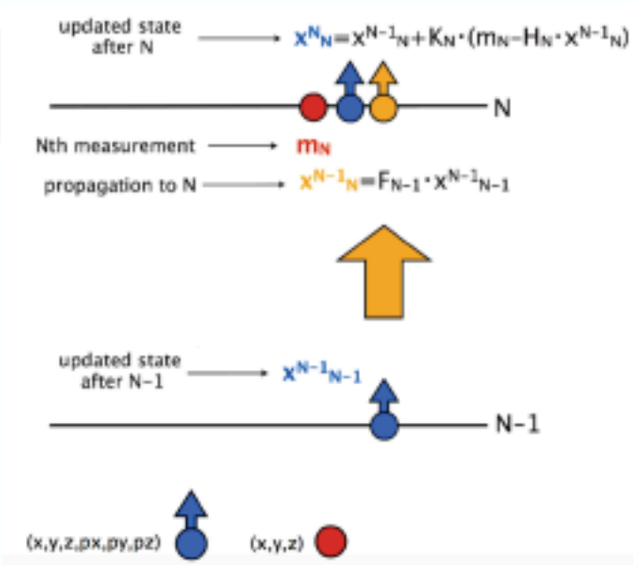
mkFit

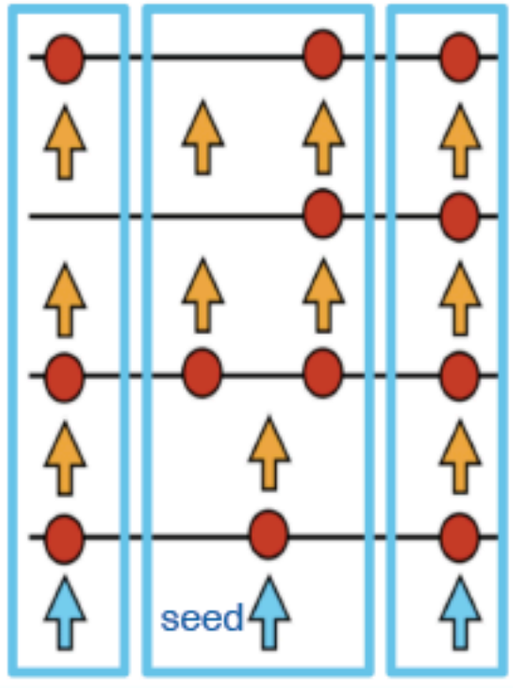
Kalman Filter

- Two step iterative process:
  - **Propagate** the track state from layer N-1 to layer N (prediction)
  - **Update** the state using the detector hit (measurement) on layer N

- Computing **challenges**:
  - Many operations with small matrices, low arithmetic intensity
  - O(2k) seeds and O(100k) hits/event @PU=70

- KF track finding is not straightforward to parallelize
  - Combinatorial algorithm: **branching** to explore many candidates
  - **Heterogeneous** environment:
    different number of hits per track and tracks per event



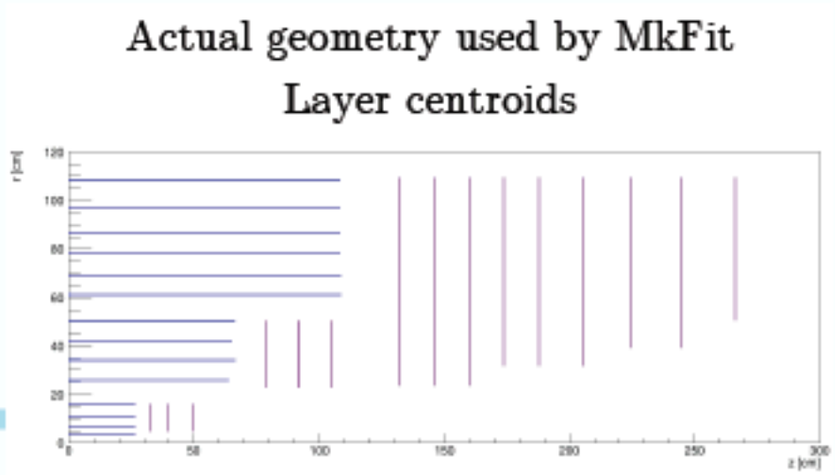


https://indico.cern.ch/event/773049/contributions/3474739

mkFit

Key Features of the Algorithm

- Kalman filter operations use **Matriplex library**: SIMD processing of track candidates
  - auto-generated vectorized code is aware of matrix sparsity
- Algorithm multithreaded at multiple levels with **TBB tasks**
  - events, detector regions, bunches of seeds
- **Lightweight description** of detector in terms of geometry, material, magnetic field
  - collapse barrel (endcap) layers at average r (z), use 3D position of hits
- **Minimize memory operations** (number and size) within combinatorial branching
  - bookkeeping of explored candidates, clone only best ranking ones at each layer (with per seed cap)



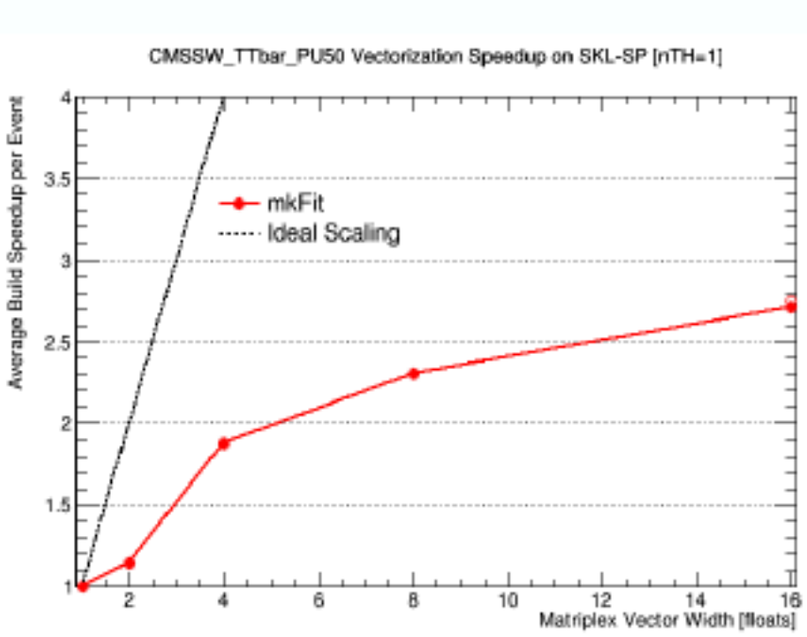
6 2019/11/07 mkFit - CHEP2019

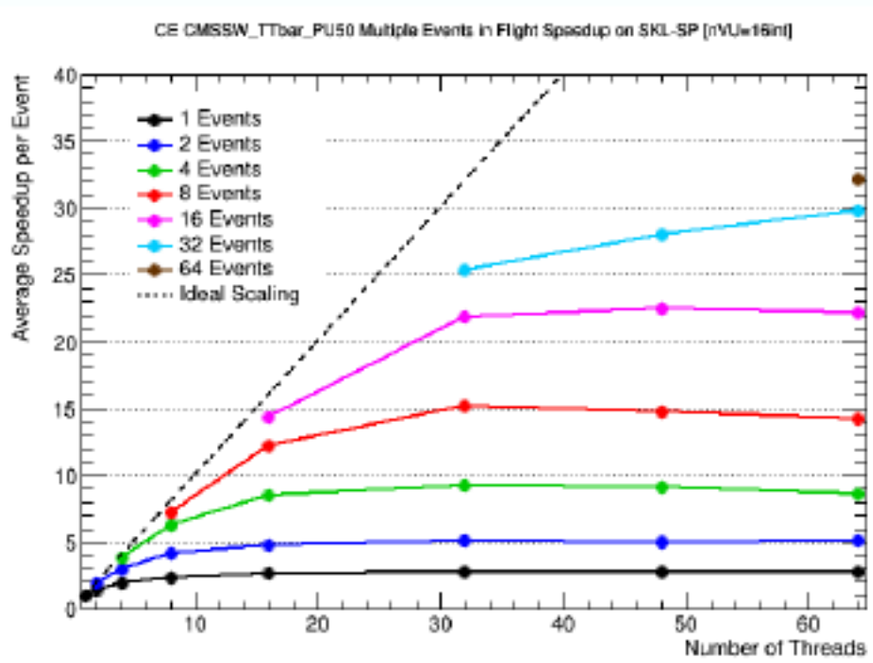
https://indico.cern.ch/event/773049/contributions/3474739

mkFit

Timing Results for Standalone Application

- Showing results on Intel Skylake Gold processor (SKL)
- Core of algorithm achieves nearly **3x** speedup from **vectorization**
  - Ahmdal's law: 60-70% of core algorithm code is effectively vectorized
- Full application achieves **30x** speedup with **multi-threading**
  - close to ideal scaling when all threads dedicated to different events





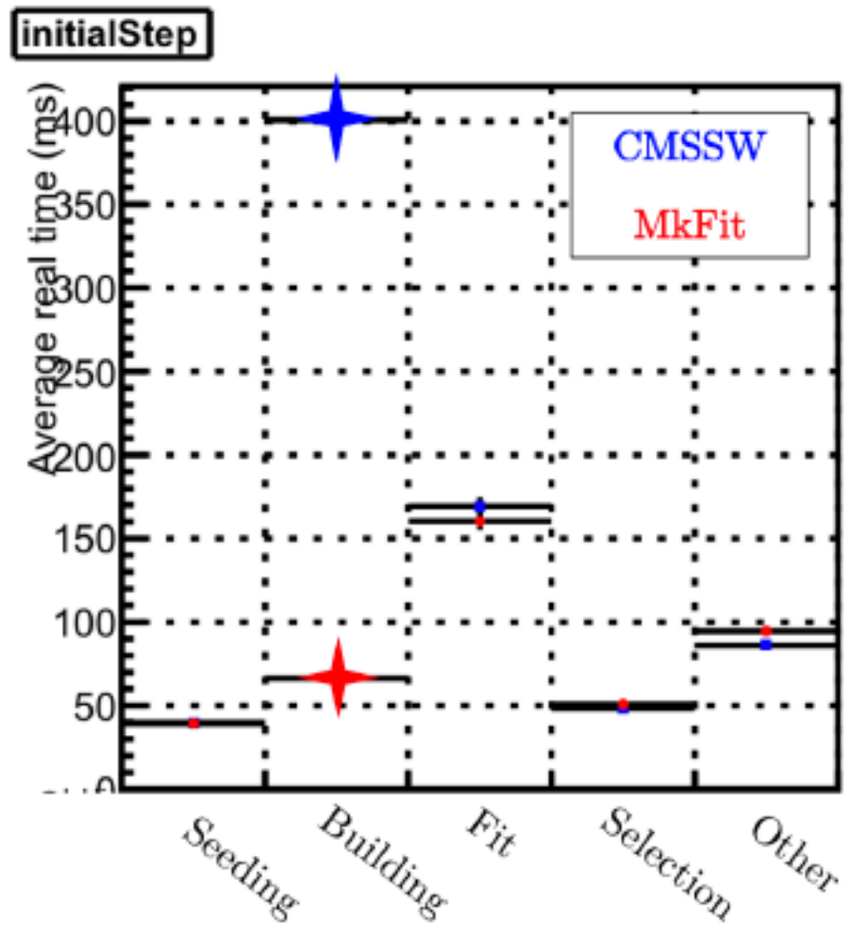
Fermilab

https://indico.cern.ch/event/773049/contributions/3474739

mkFit

Timing Performance of Initial Iteration

- **Single-thread performance** on Intel SKL
  - use ttbar events with <PU>=50
- Speedup of **6.2x** compared to CMSSW
  - track building is not the slowest component anymore!
- Data format **conversions** between CMSSW and mkFit account for **~25%** of mkFit time
  - larger speedup possible if data formats are harmonized
- Here mkFit is compiled with icc and AVX-512
  - with gcc speedup reduces to ~2.5x



Fermilab

10 2019/11/07 mkFit - CHEP2019

https://indico.cern.ch/event/773049/contributions/3474739

CMS HLT Phase-2

But How To Take The Data?

CMS detector	LHC Run 2	HL-LHC Phase-II	
Peak ⟨PU⟩	60	140	200
L1 accept rate (maximum)	100 kHz	500 kHz	750 kHz
Event Size	2.0 MB	5.7 MB	7.4 MB
Event Network throughput	1.6 Tb/s	23 Tb/s	44 Tb/s
Event Network buffer (60 s)	12 TB	171 TB	333 TB
HLT accept rate	1 kHz	5 kHz	7.5 kHz
HLT computing power	0.5 MHS06	4.5 MHS06	9.2 MHS06
Storage throughput	2.5 GB/s	31 GB/s	61 GB/s
Storage capacity needed (1 day)	0.2 PB	2.7 PB	5.3 PB

More details on the DAQ system side: go and see M. Dobson's talk!

- Baseline DAQ and HLT systems
- Evolution of HS network fabrics
- Performance of general-purpose CPUs
- DAQ and Timing Hub (DTH) board

Trigger and DAQ system

- Balance efficiency, rates, event size and timing
- Level 1 Trigger (hardware) + High-Level Trigger (software)
- Baseline: all data saved in RAW format and promptly reconstructed

7.5 kHz assumptions

- 750 kHz L1T output in 12.5 μs
  - L1 Track-Trigger and Particle Flow
- 1/100 reduction at HLT (!)

HLT farm implications

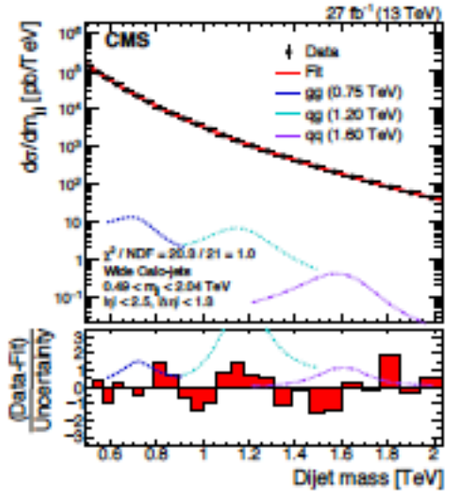
- 18 X more CPU power
- 27 X more storage

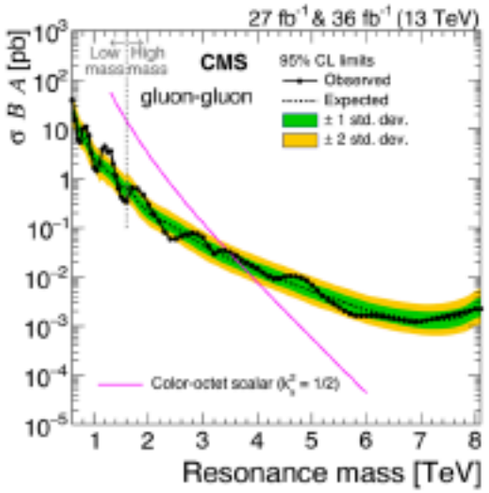
4

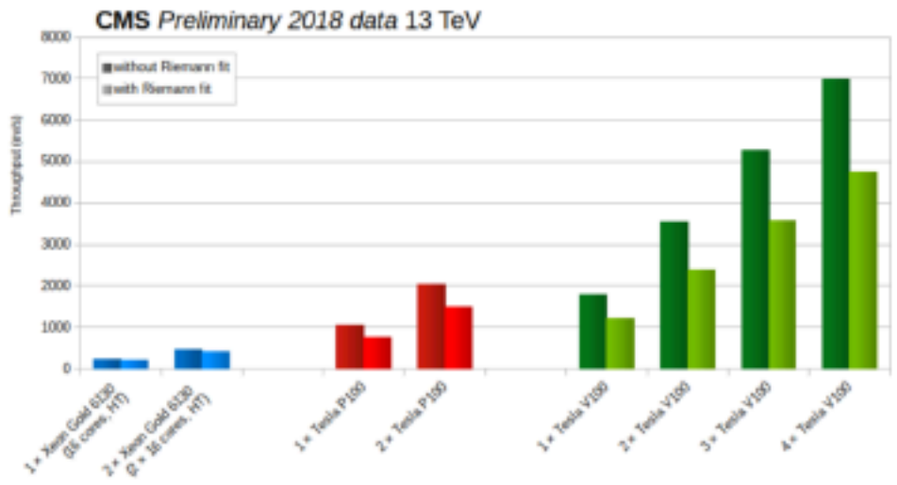
https://indico.cern.ch/event/773049/contributions/3474308

CMS HLT Phase-2

High-Level Trigger – Alternative Approaches







Scouting: shrink size, increase rate

- HLT calo: simpler objects, limited by L1 thresholds and rate
- HLT PFLow: complex objects, limited by HLT timing
- L1 – new proposal!
  See E. Meschi's talk for more details!

Heterogeneous architecture

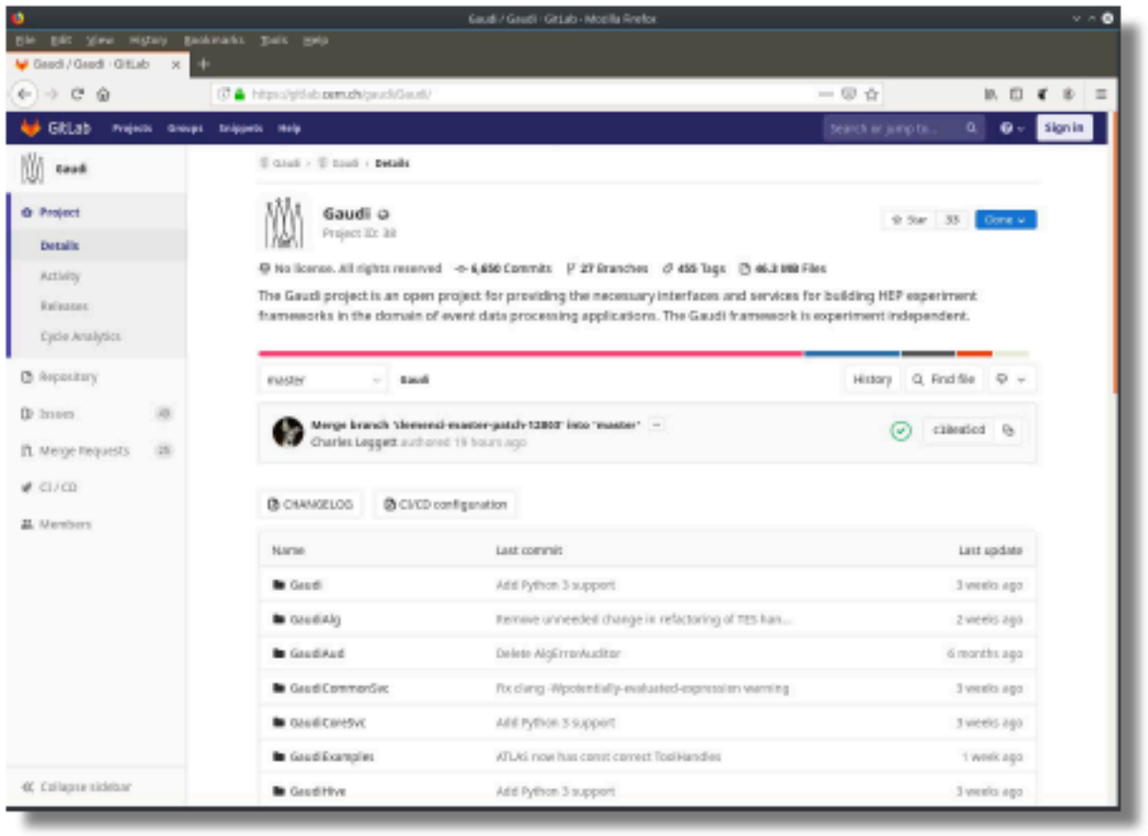
- GPUs, ARM + GPUs, …
- Currently: pixel tracking with GPUs
  - To be deployed at Run 3
- See A. Bocci's talks for more details!

11

https://indico.cern.ch/event/773049/contributions/3474308

ATLAS

Gaudi / Athena



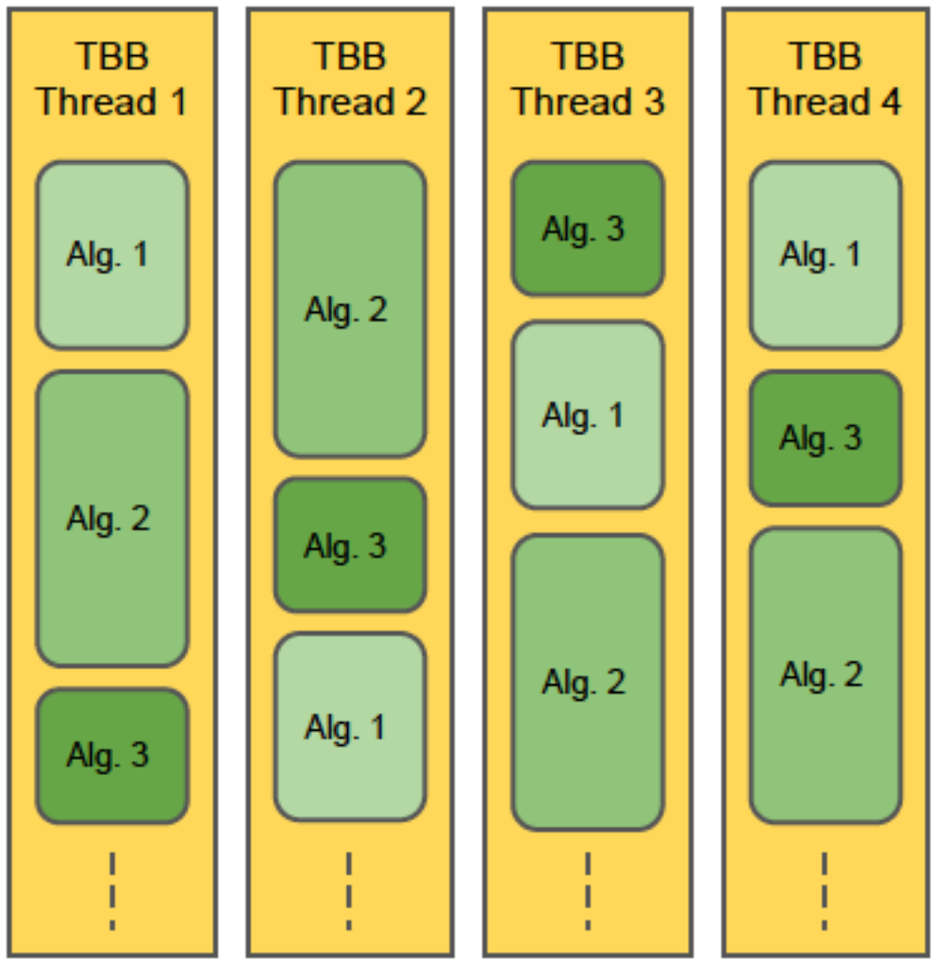
- ATLAS and LHCb share Gaudi as the basis of their software frameworks
  - ATLAS calls its own framework, built on top of Gaudi, Athena
- The framework defines "algorithms" as the base unit of execution
  - Classes that have an `execute(...)` function, which performs some data processing with the help of various "services" and "tools"

6

https://indico.cern.ch/event/773049/contributions/3473268

ATLAS MT and GPU in reco

Task Scheduling in AthenaMT



- Athena (Gaudi) uses TBB to execute algorithms on multiple CPU threads in parallel
  - The framework's scheduler takes care of creating TBB tasks that execute algorithms, at the "right times"
- The goal, of course, is to fully utilise all CPU cores assigned to the job, but not to use more
  - So any offloading needs to thoughtfully integrate into this infrastructure

8

- **A scheduler for offloading work on GPU is in development**

https://indico.cern.ch/event/773049/contributions/3473268

Trigger Level analysis

Trigger-level analysis

To generically probe the entire range of EW–TeV dijet resonances with the full statistical power of the data, we need to work around all three trigger limitations (L1, HLT CPU, HLT rate).

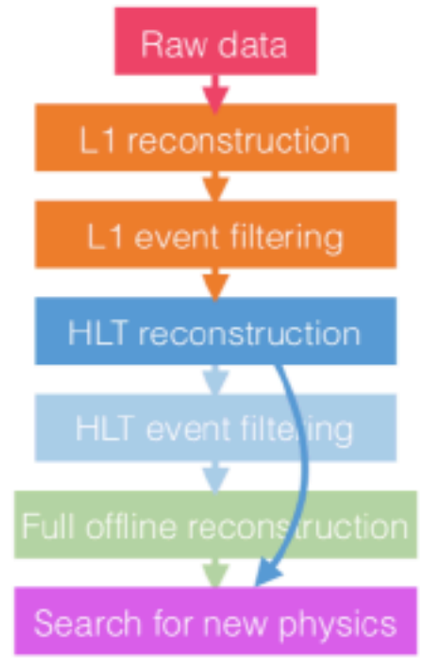
This can only be done within the trigger itself, i.e. trigger-level analysis (TLA).

Difference between L1 and HLT thresholds (200–440 GeV, shown earlier) suggests a first step for Run 2: improve (already good) and **analyze the HLT jet reconstruction at the L1A rate**; throw out the full data.

This technique also employed at LHCb (turbo stream) and CMS (data scouting).

TLA stream records **only HLT objects** (jet four-vectors, jet ID and calibration variables, etc.) for **specific L1A**.

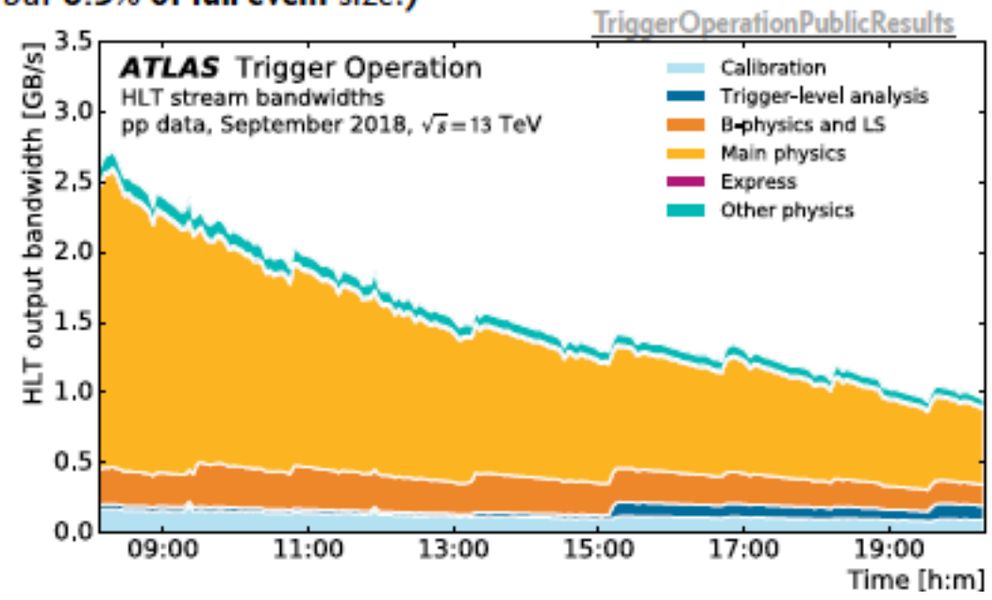
Throw out other information (e.g. **no tracking information kept** in Run 2, but **0.5% of full event** size.)

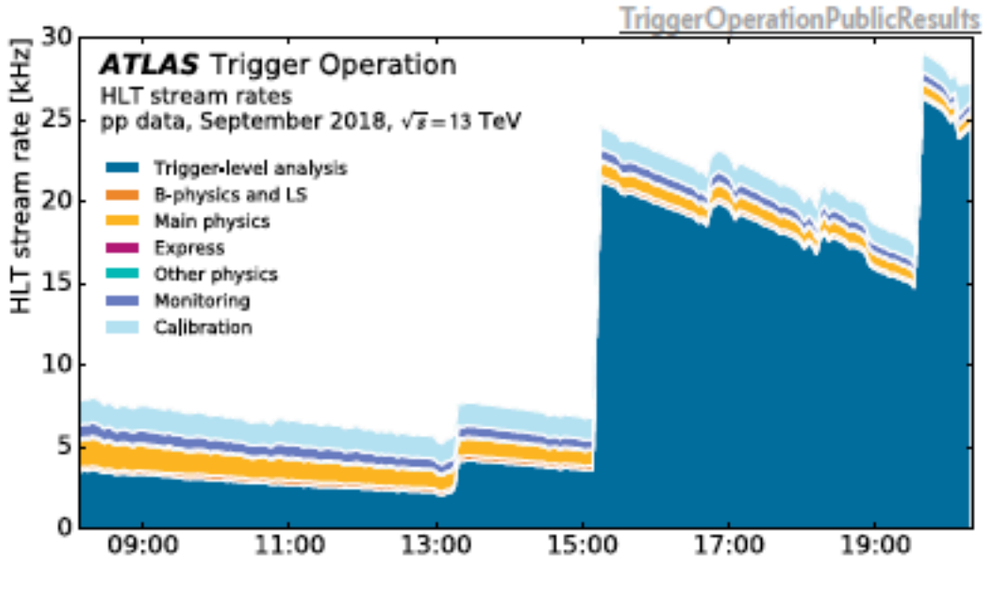


~3 kHz

~~0.025 kHz~~

~3 kHz





https://indico.cern.ch/event/773049/contributions/3474303

Outlook for Run-3 and HL-LHC

Run 3 will bring several relevant improvements to the trigger hardware and software

Better HLT object calibration and resolution

Possibility of particle-flow jets for Run-3 (better jet resolution at low p_T)

Pile-up mitigation for better control of lower-momentum (~<100 GeV) jets

Possible software tracking for rejection of pile-up for lower-momentum jets; additional objects

Improved software flexibility for **partial-event readout**

Improved performance with **new L1 hardware**

HL-LHC

Powerful first-stage trigger capabilities with L0 Global Trigger upgrade and HLT Hardware Track Trigger

Storage and computing pressures increase

but TLA also offers a solution, at least for some types of standard physics analyses

For details on these and further ideas, see also ATL-DAQ-PUB-2017-003 and related HSF-CWP-2017-01.

10

https://indico.cern.ch/event/773049/contributions/3474303

LHCb trigger GPU

The LHCb Upgrade

The LHCb detector and Data Acquisition system will be upgraded to prepare for a new data taking period in 2021. The design of the upcoming trigger[1] will be challenging due to two factors:

- LHCb will remove its hardware level trigger, turning to a full-software trigger
- Luminosity will increase to $2 \times 10^{33} \mathrm{cm}^{-2} \mathrm{s}^{-1}$

LHCb will filter data at a rate of 40 Tbit/s in a software trigger

[1] *LHCb Trigger and Online Upgrade Technical Design Report*, https://cds.cern.ch/record/1701361

https://indico.cern.ch/event/773049/contributions/3473255

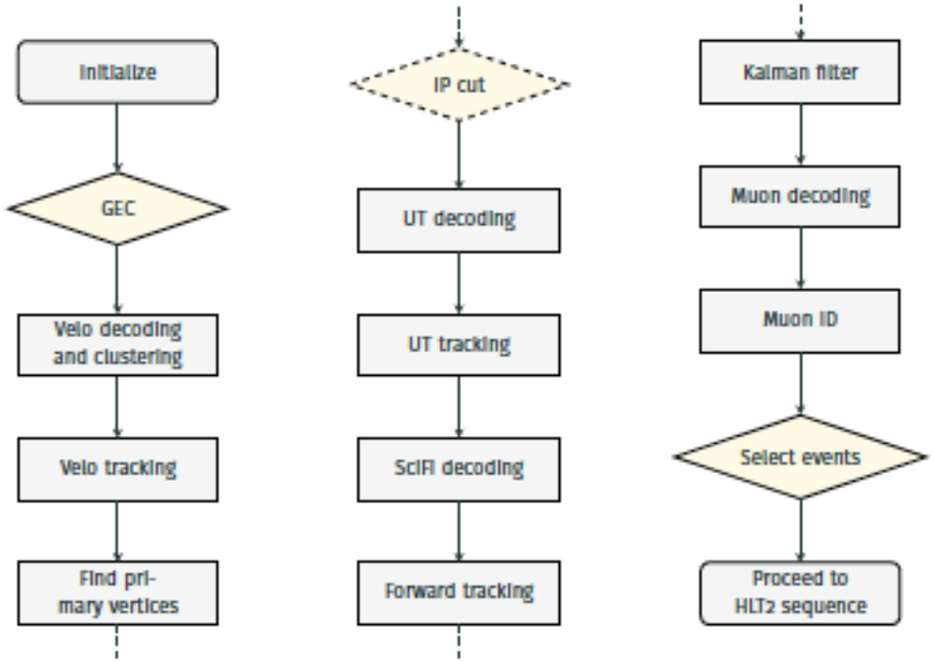
- **NB: pileup in Run3 is ~6, compared to ~1.1 in Run1,2**

LHCb trigger GPU

High Level Trigger 1 at LHCb

The first stage of software trigger, also known as *High Level Trigger 1*, is a critical stage of the software reconstruction. It must make a decision in near-time over all input data, at the collision rate.

The entire HLT1 involves the decoding, clustering and track reconstruction of all tracking detectors at LHCb, as well as the Kalman filter, PV finder and trigger decision algorithms.



5

https://indico.cern.ch/event/773049/contributions/3473255

LHCb trigger GPU

A GPU HLT1 for the LHCb Upgrade

We are proposing to run the entire LHCb HLT1 on GPUs. Our proposal features:

- *Full software HLT1 sequence* — The baseline HLT1 programme has been fully implemented.
- *Scalability and versatility* — Growing codebase encompassed by framework.
- *Compatibility* – The codebase compiles across architectures.
- *Compact solution* — Strategic placement in the Event Builders allows for cost savings.

6

https://indico.cern.ch/event/773049/contributions/3473255

LHCb trigger GPU

The Allen framework

The *Allen* framework is a compact, scalable and modular framework, built for running the LHCb HLT1 on GPUs.

Requirements

- A C++17 compliant compiler, boost, zeromq
- CUDA v10.0

Features

- Configurable static sequences.
- Pipelined stream sequence.
- Custom memory manager, no dynamic allocations, SOA datatypes.
- Built-in validation with Monte Carlo.
- Optional compilation with ROOT for generation of graphs.
- Integration with Gaudi build system.
- Cross-architecture compatibility.

7

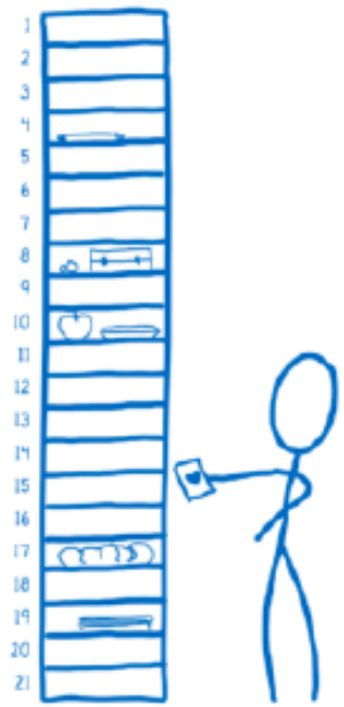
https://indico.cern.ch/event/773049/contributions/3473255

LHCb trigger GPU

Memory management

In Allen, we allocate memory at the **startup** of the application. A custom memory manager assigns memory segments on demand.

- All data dependencies and memory assignments are resolved at compile-time.
- There are no dynamic memory allocations.
- GPU memory is plenty for the Allen use case.

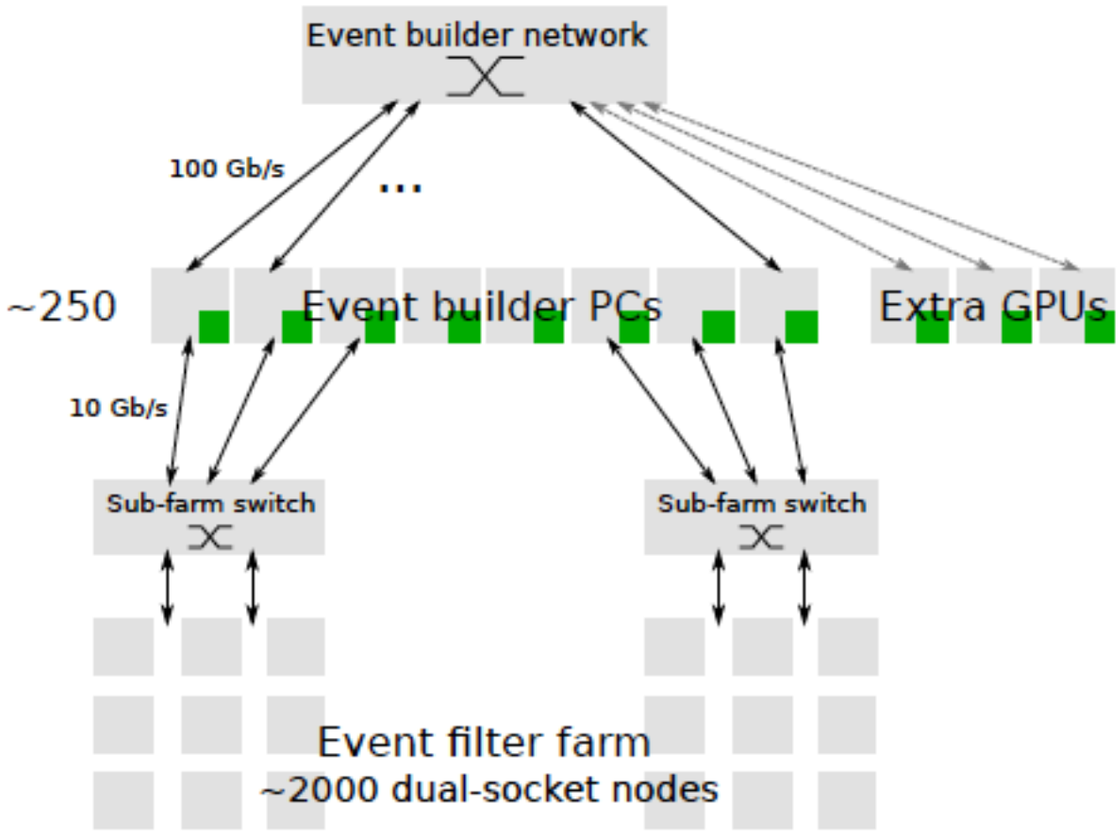


9

https://indico.cern.ch/event/773049/contributions/3473255

LHCb trigger GPU

Event builders with GPUs



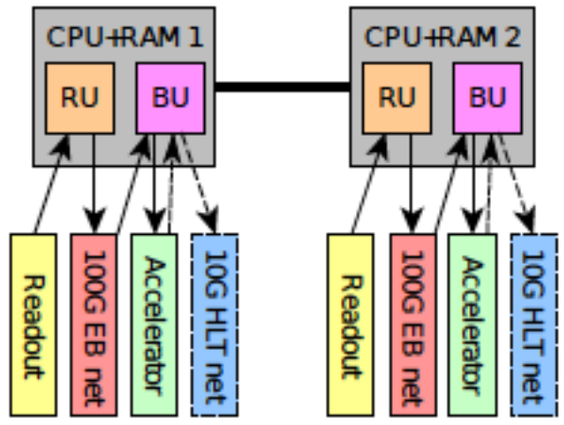


Figure 2: GPU-equipped event builder PC.

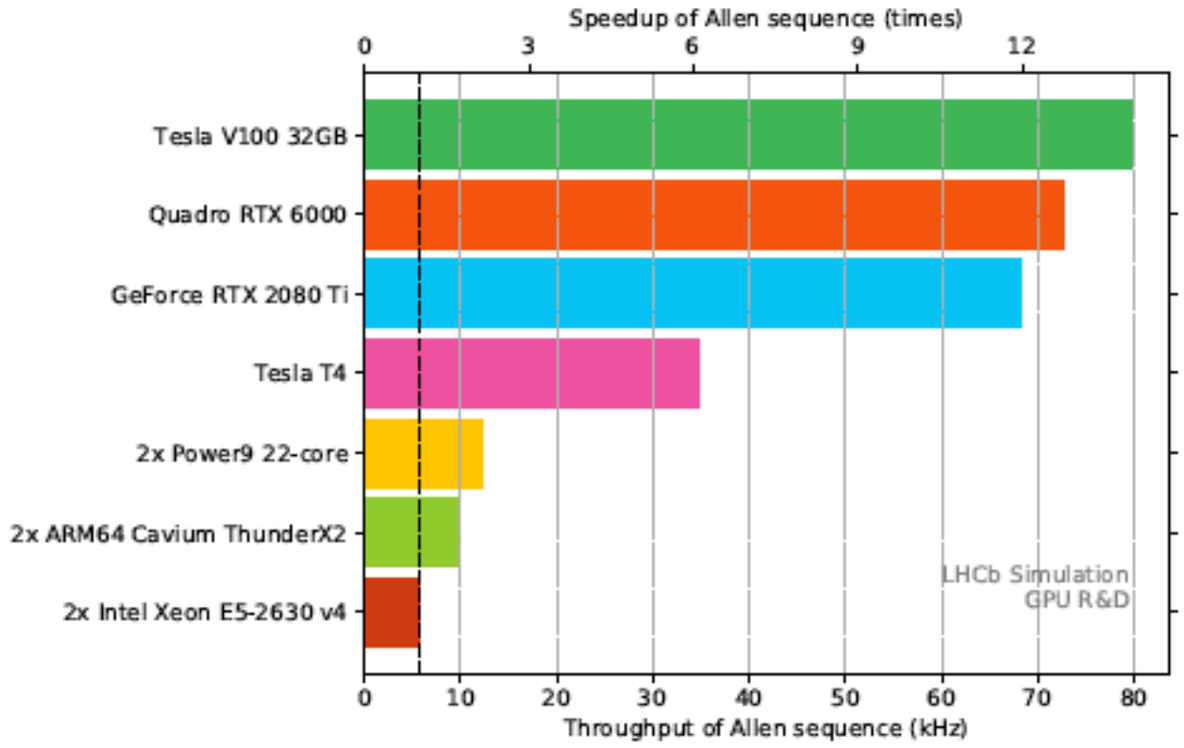
12

https://indico.cern.ch/event/773049/contributions/3473255

LHCb trigger GPU

Target processing rate

In order to be able to perform the HLT1 filter inside the event builder with GPUs, the full throughput of collisions must be processed in near-time.



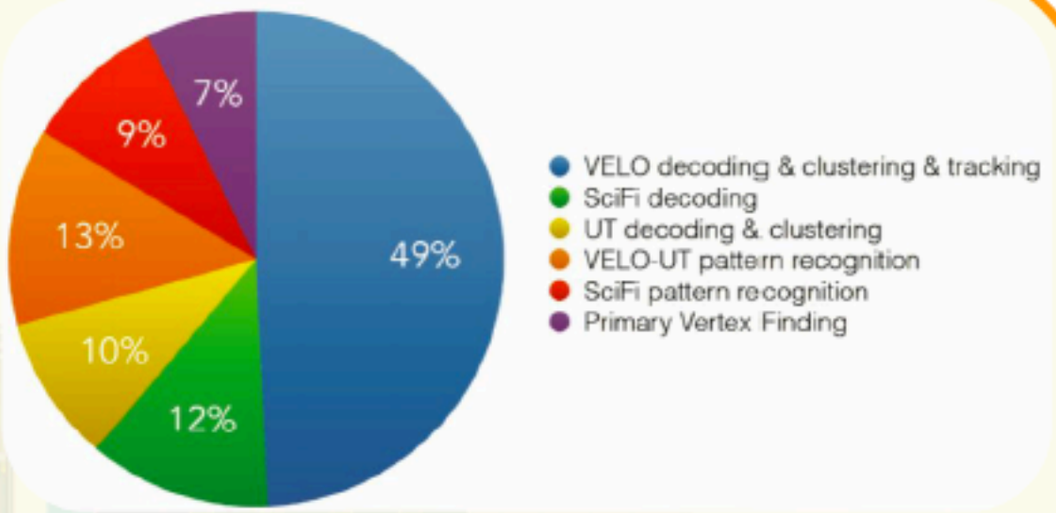
13

https://indico.cern.ch/event/773049/contributions/3473255

LHCb tracking with FPGAs

LHCb-Upgrade tracking sequence (HLT1)

- Majority of time-budget spent in repetitive and parallelizable tasks rather than refined physics selections
- VELO tracking: first and fundamental step of the HLT1 tracking sequence
- Also the **most time-consuming** one!



HLT1 CPU-time usage
(~35 MHz total throughput on a 1000-server Farm)

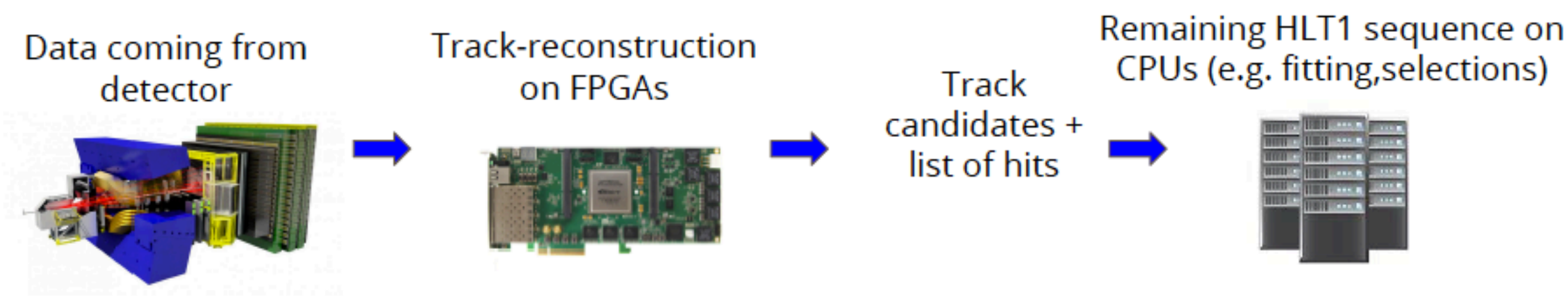
https://indico.cern.ch/event/773049/contributions/3474316

LHCb tracking with FPGAs

Tracking with FPGAs

➡ Idea: **offload parts of this VELO reconstruction to an FPGA-based system**

- FPGA is the appropriate technology: aim for high bandwidth and low latency, comparable with that of other elements in the detector DAQ
- Save the flexibility of CPUs for more intelligent jobs if successful
  - Example: saved time can be used to run part of HLT2 (the second stage fo software trigger)
- This talk: status R&D work for LHCb-Upgrade and performance obtained

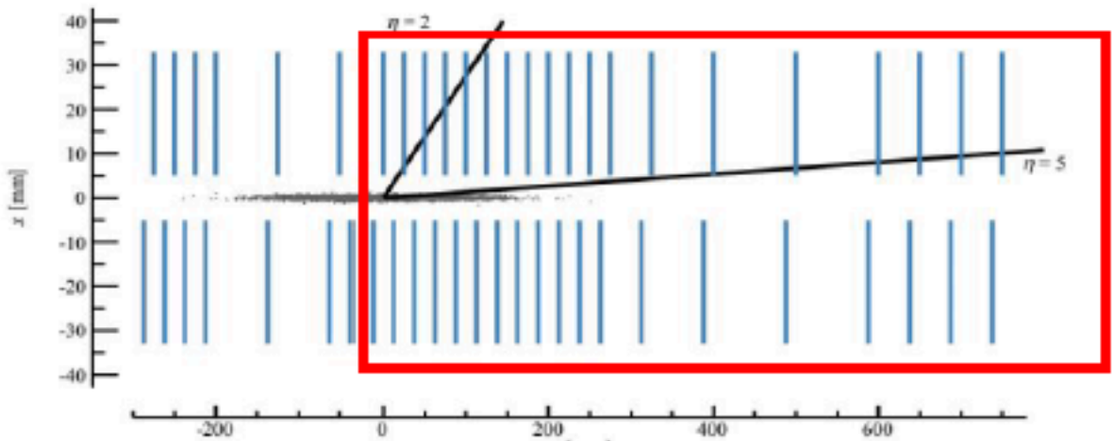


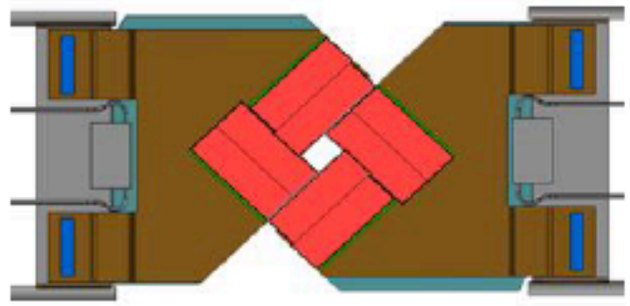
G.Tuci, 05/11/19 Reconstruction of track candidates using FPGAs 5

https://indico.cern.ch/event/773049/contributions/3474316

LHCb tracking with FPGAs

The VELO detector





- Composed by 52 silicon pixel modules, arranged in 26 layers
  - Each module is read by a separate readout unit
- In the current studies we use data from layers 8 to 26 → **38 DAQ board**
  - Aim at forward-going tracks only. Other layers used mainly to optimize primary vertex precision

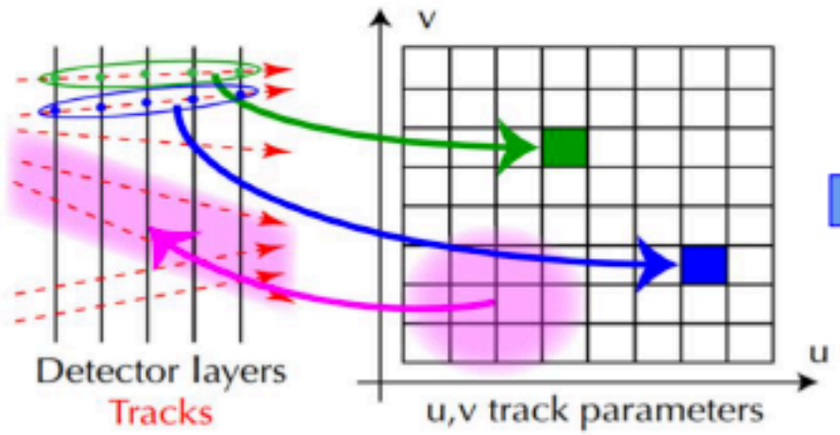
G.Tuci, 05/11/19 Reconstruction of track candidates using FPGAs 7

https://indico.cern.ch/event/773049/contributions/3474316

LHCb tracking with FPGAs

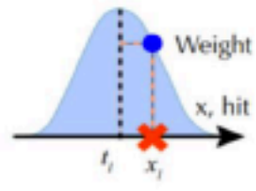
The "artificial retina" algorithm

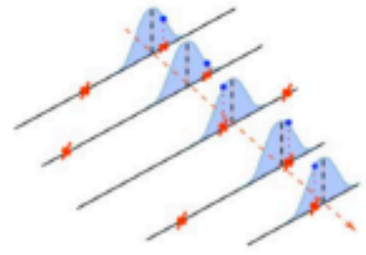
NIMA 453 (2000) 425-429

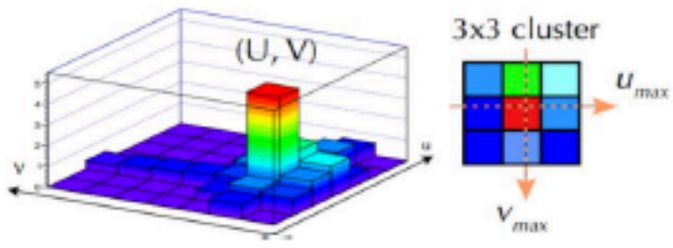


Parameter space divided into cells.
Each cell→ pattern.
Receptors: interception of pattern tracks with detector layers.

Distance hit- receptor is calculated.
A weight is assigned to each distance.
For each cell all the weights are summed (excitation level).







Search of local maxima over a threshold → reconstructed tracks

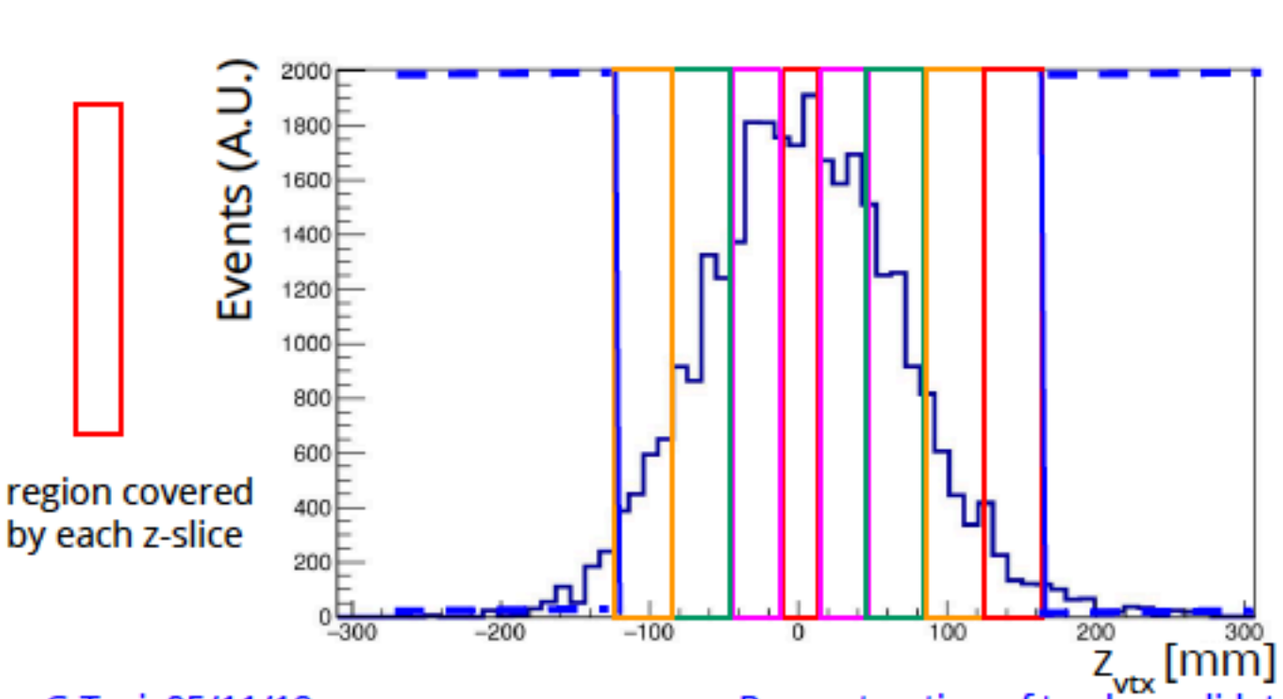
Everything is executed in parallel!

https://indico.cern.ch/event/773049/contributions/3474316

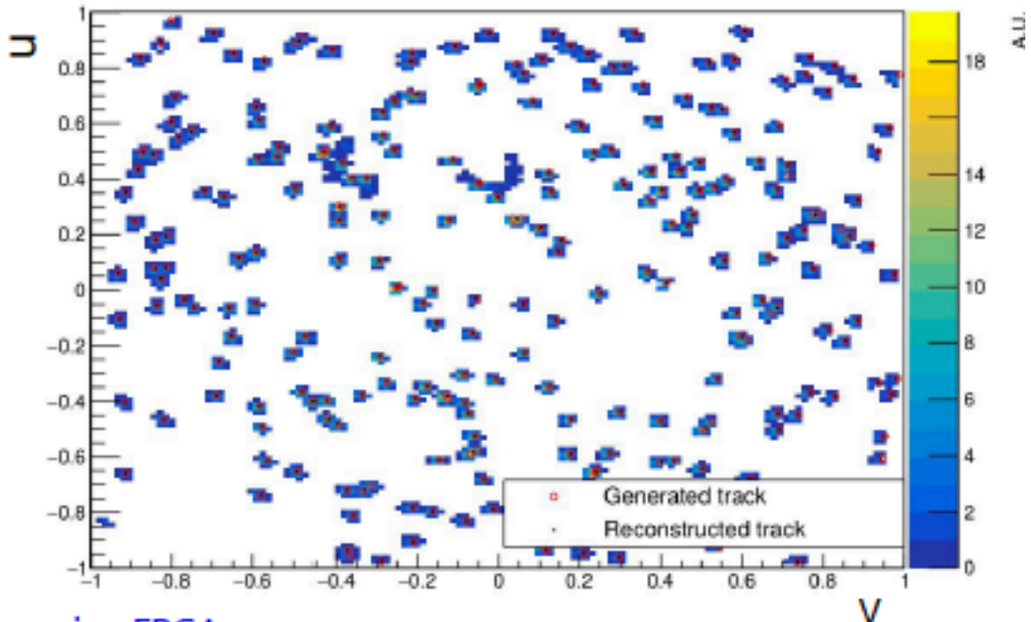
LHCb tracking with FPGAs

Segmentation of parameter space

- u,v are the x,y coordinates of intercept with a fixed plane
- Algorithm based on a 2D space
  - segment track space in 10 z slices * 10000 cells each (100k total cells)
  - Fits well FPGA logic available in current COTS PCIe cards.



Distribution of excitation levels for one slice



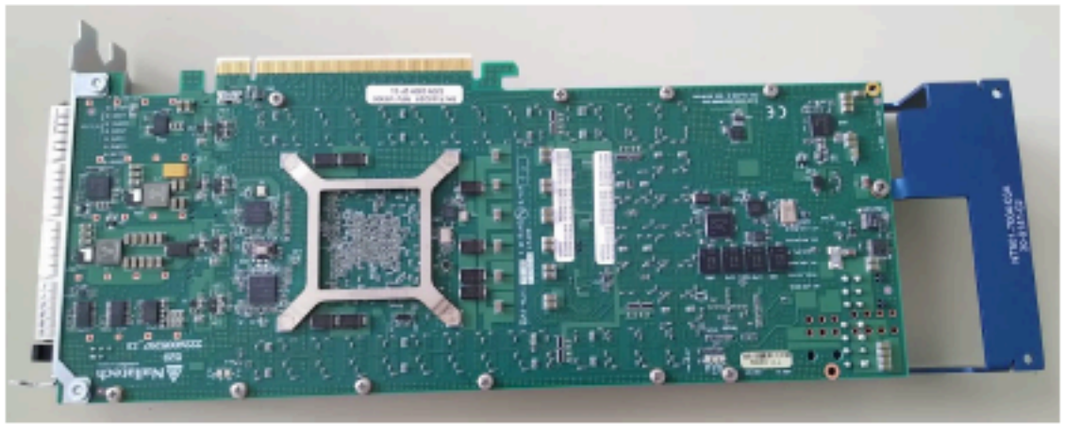
https://indico.cern.ch/event/773049/contributions/3474316

LHCb tracking with FPGAs

Implementation status

- Old hardware prototype of generic 6-layer tracker already demonstrated throughput exceeding 30 MHz, but ***need to specialize for the VELO*** PoS(TWEPP-17) 136
- Logic resources: estimate to fit comfortably in the **38 current COTS FPGA cards,** needed to match the readout boxes of the VELO

Commercial board equipped with Altera Stratix 10 FPGA is under testing and it is a potential candidate for the final system →

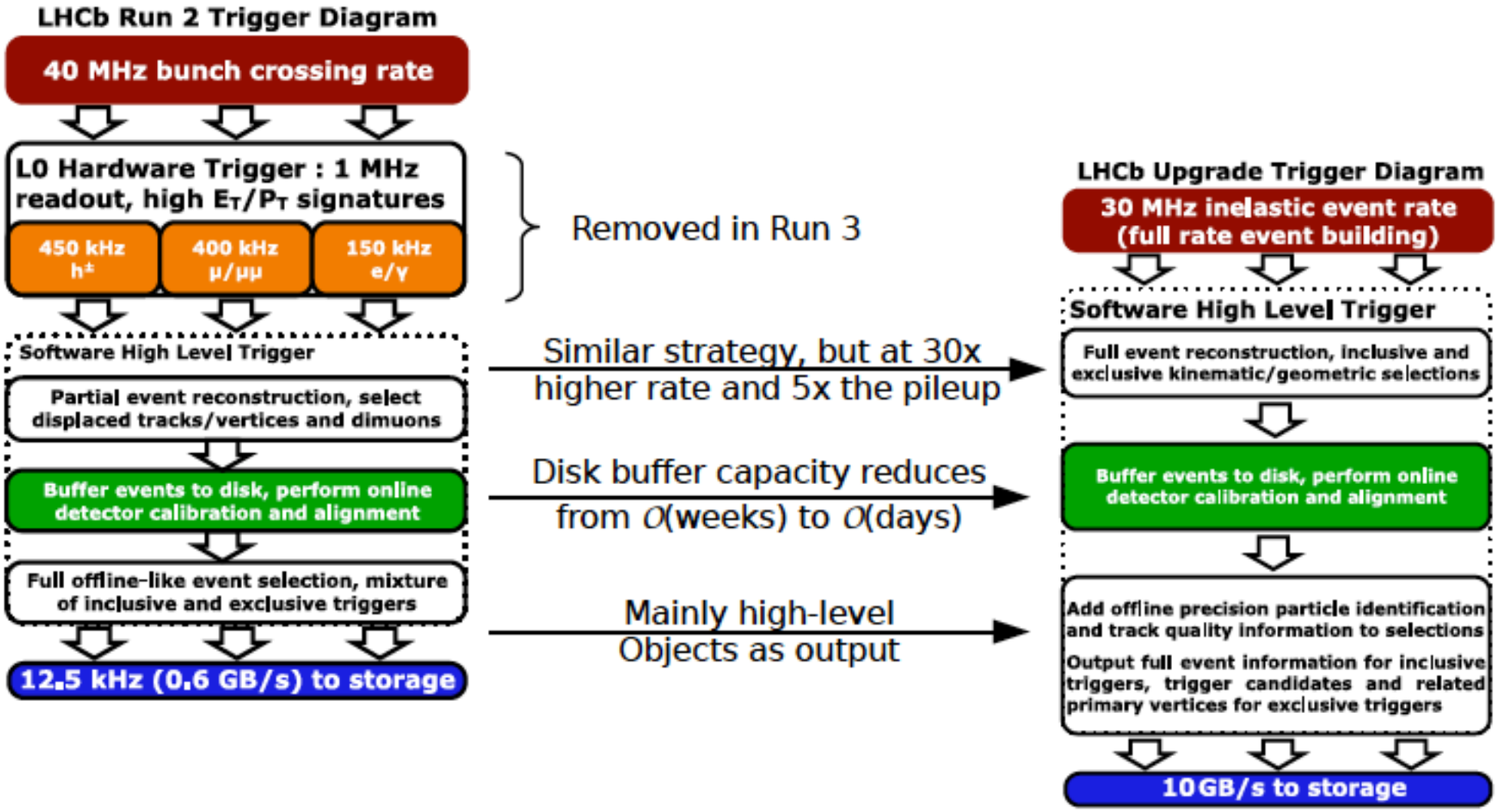


G.Tuci, 05/11/19 Reconstruction of track candidates using FPGAs 17

https://indico.cern.ch/event/773049/contributions/3474316

LHCb trigger upgrade

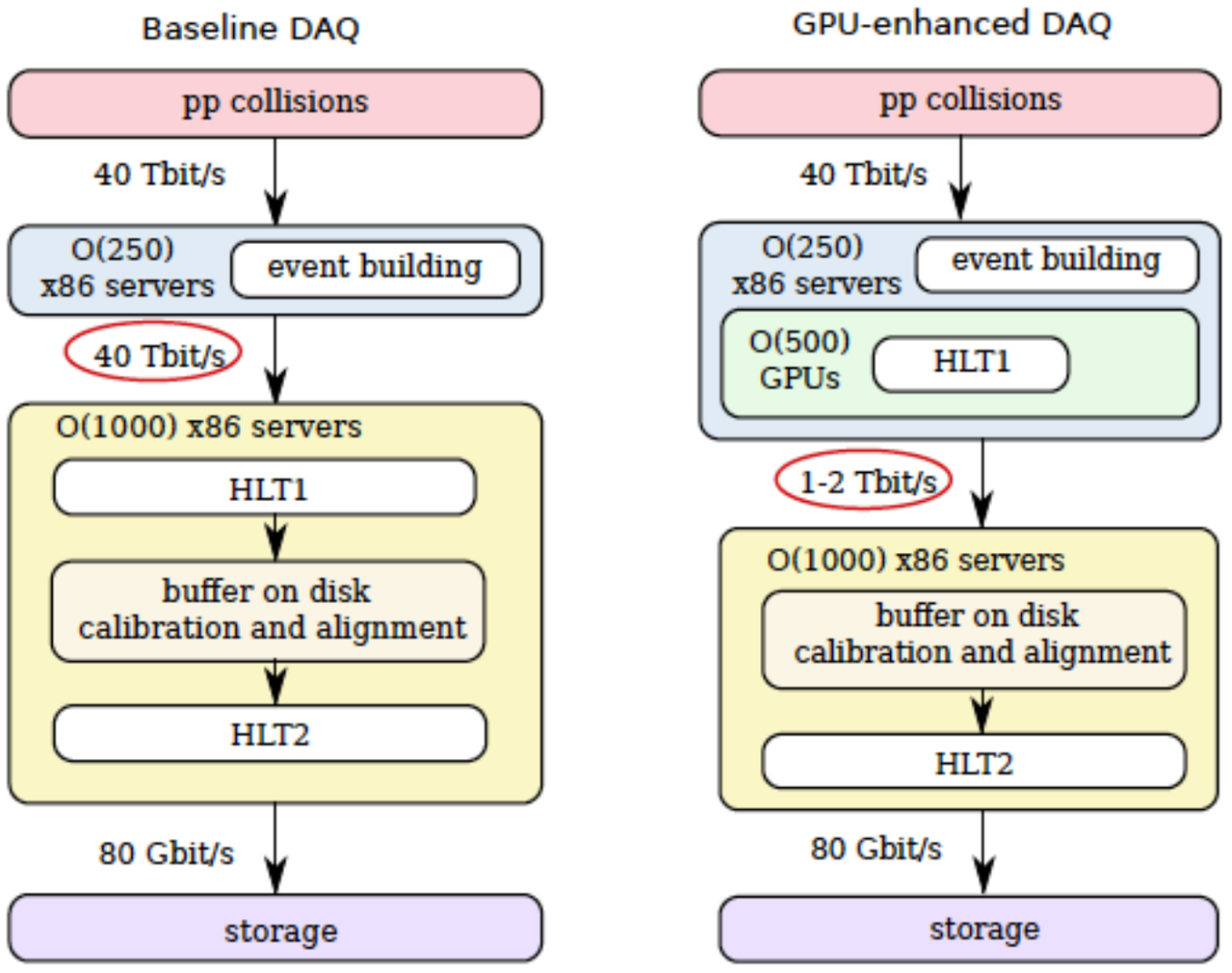
Trigger upgrade for Run 3 (2021)



https://indico.cern.ch/event/773049/contributions/3474298

LHCb trigger upgrade

Where to place the GPUs?



GPUs naturally integrate into LHCb's DAQ

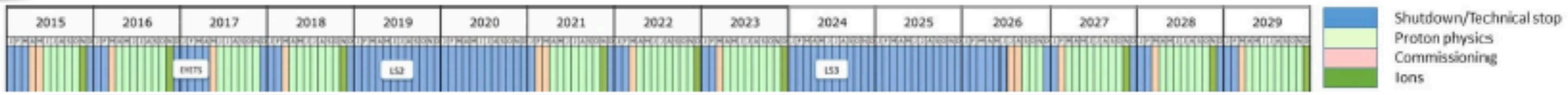
If HLT1 can run on 500 GPUs
→ Save money on network
→ Buy GPUs instead

16

https://indico.cern.ch/event/773049/contributions/3474298

ALICE towards Run 3, 4

Run2 → Run3 and 4



Run2: $\mathcal{L}_{int}^{Pb-Pb}$ = 1.0 nb^{-1}

Run3: $\mathcal{L}_{int}^{Pb-Pb}$ = 6.0 nb^{-1}

Run4: $\mathcal{L}_{int}^{Pb-Pb}$ = 7.0 nb^{-1}

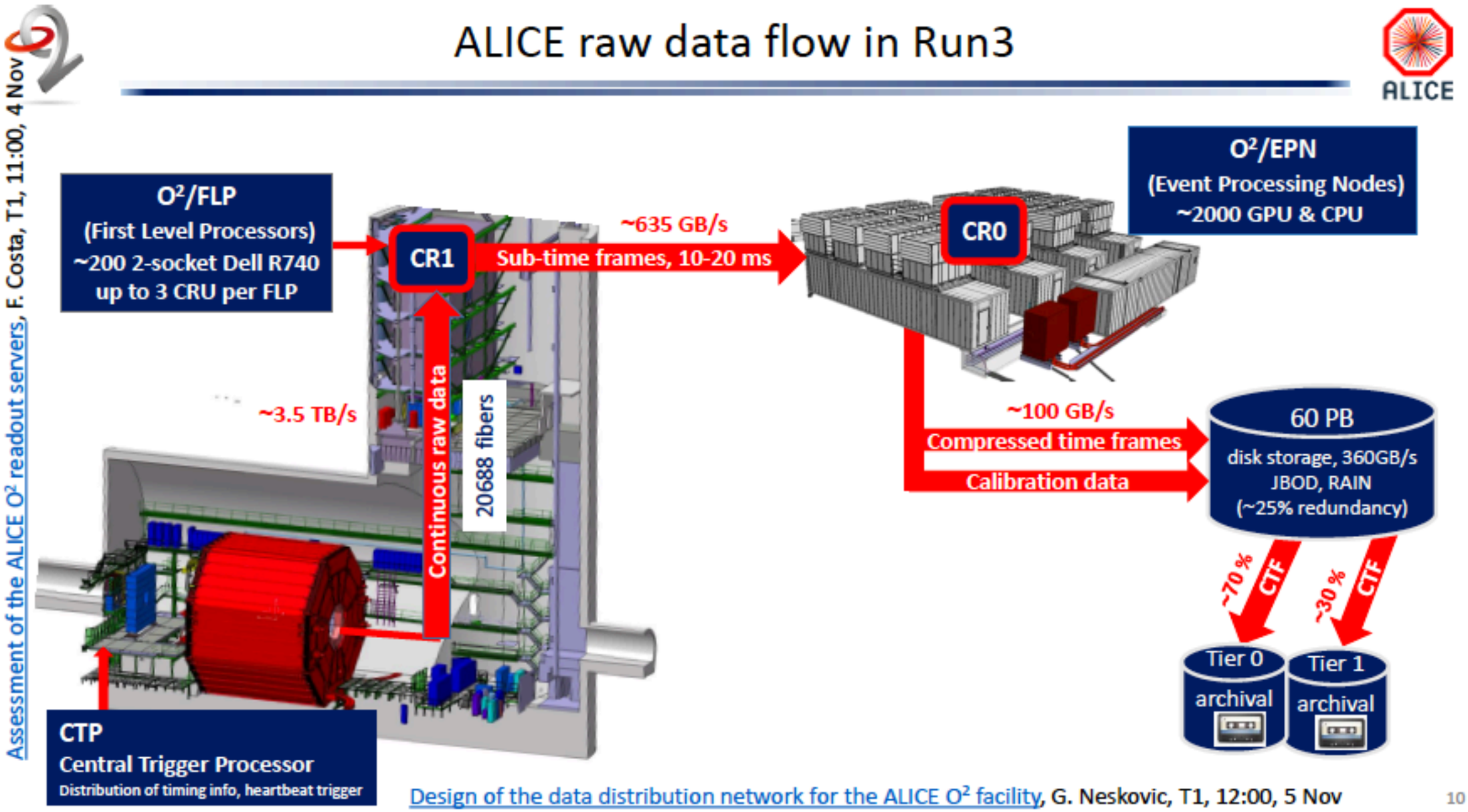
- In Run2 ALICE operated at Pb-Pb interaction rates ~7-10 kHz (inspected ~1 nb^{-1}) with trigger rate < 1 kHz
- LHC plans to deliver 50 kHz Pb-Pb interaction rate after LS2
- ALICE plans for Run3&4: collect 13 nb^{-1} of Pb-Pb collisions at 5 TeV (of which 3 nb^{-1} with reduced field)
- Main limitations to work at these rates:
  - Principal tracking detector, TPC has ~90 μs drift time + at least ~200 μs gating grid to collect the ion backflow
    → trigger rate limited to ~3 kHz (< 1 kHz accounting for bandwidth)
- At high multiplicities (dN/dη ~ 2000 + pile-up) very low S/B for rare probes: dedicated (HLT) trigger is not realistic

➡ Use continuous readout at least for TPC (no gating grid), increase bandwidth
Read out all events, store compressed data and inspect all events offline

5

https://indico.cern.ch/event/773049/contributions/3581368

ALICE towards Run 3, 4



https://indico.cern.ch/event/773049/contributions/3581368

ALICE tracking in Run 3

Tracking in ALICE in Run 3

ALICE

- **Bulk of computing workload:**

Synchronous	Asynchronous
• >90% TPC tracking / compression	• TPC among largest contributors
• Low load for other detectors	• Other detectors also significant

- **ALICE GPU processing strategy**

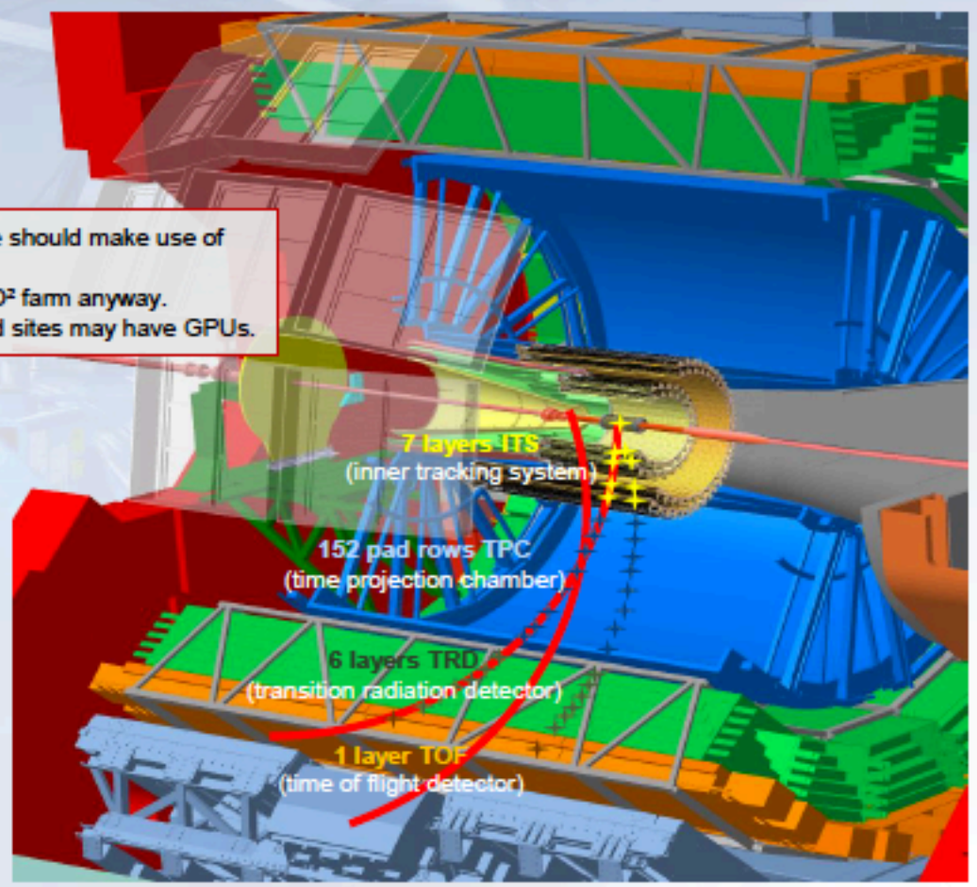
Baseline solution
(almost available today):
TPC + part of ITS tracking on GPU

– **Mandatory** solution to keep up with the data rate online.
– **Defines** number of servers / **GPUs**.

Optimistic solution
(what could we do in the ideal case):
Run most of tracking + X on GPU.

– Extension of baseline solution to make best use of GPUs.
  – Ideally, **full barrel tracking** without ever leaving the GPU.
  – In the end, we will probably be somewhere in between.

Asynchronous phase should make use of the available GPUs.
- Available in the O² farm anyway.
- Future HPC / grid sites may have GPUs.



5.11.2019 David Rohr, drohr@cern.ch 3 / 9

https://indico.cern.ch/event/773049/contributions/3581368

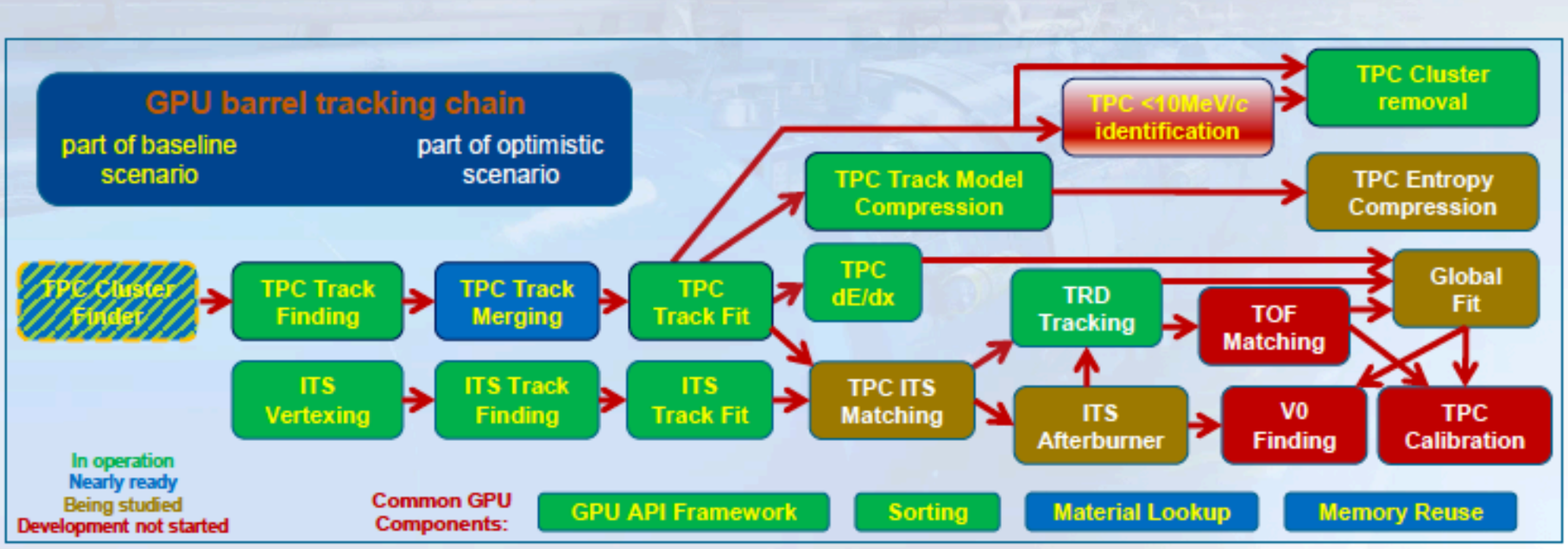
ALICE tracking in Run 3

Reconstruction steps on GPU (Barrel Tracking)

ALICE

- **Status of reconstruction steps on GPU:**
  - All TPC steps during synchronous reconstruction are **required** on the GPU.
  - Synchronous ITS tracking and TPC dE/*dx* in good shape, thus considered **baseline** on the GPU.
  - Remaining steps in tracking chain part of **optimistic scenario**, being ported step by step to GPU.
    - Porting order follows topology of chain, to avoid unnecessary data transfer for ported steps – current blocker is **TPC ITS matching**.

TRD tracking / TPC calibration: see poster of Ole Schmidt
Space point calibration of the ALICE TPC with track residuals



5.11.2019 David Rohr, drohr@cern.ch 4 / 9

https://indico.cern.ch/event/773049/contributions/3581368

ALICE tracking in Run 3

Memory requirements

Work in Progress

- **ALICE reconstructs timeframes (TF) independently (~10 – ~20 ms; 128 – 256 orbits; ~500 – ~1000 collisions).**
  - One TPC drift time of data not reconstructible at TF border (~ 90 us) → < 1 % of statistics lost (< 0.5 % for 20 ms).
  - Timeframe should fit in GPU memory. If not, could use kind of ring buffer, or reduce TF length to 128 orbits.
  - Trying to avoid the ring buffer approach, could be added later if needed.
- **Custom allocator: grabs all GPU memory, gives out chunks manually, memory will be reused when possible.**
  - Classically: reuse memory between events, collisions are not that large.
  - ALICE reuses memory between different algorithms in a TF, possibly also between independent collisions.
  - Some memory must persist during timeframe processing.

Persistent data — *Non-persistent scratch data for algorithms* — *Non-persisting input data*

TPC Hits 1	TPC Hits 2	TPC Hits 3	TPC Hits 4	ITS Hits	TPC Tracks	ITS Tracks	Matches	Memory	TPC Raw 2	TPC Raw 1

- **Estimated maximum memory needed during important for 10 ms TF (*2 for 20 ms):**
  - TPC Cluster finder: ~ 3 GB (+ input / scratch data, which is pipelined)
  - TPC Transformation: 12.1 GB
  - TPC Sector tracker: ~ 14.6 GB (including persistent memory from previous steps)
  - TPC Merger / track fit: 14.1 GB
  - TPC Compression: 12.9 GB
    - Later steps do not scale their scratch memory with TPC input → less memory intensive.

→ **16 GB GPU will suffice for 10 ms TF (unclear for 12 GB after optimizations).**

- 8 GB insufficient for 10 ms TF, 20 ms TF needs 32 GB, alternatively ring buffer.

5.11.2019 David Rohr, drohr@cern.ch 6 / 9

https://indico.cern.ch/event/773049/contributions/3581368

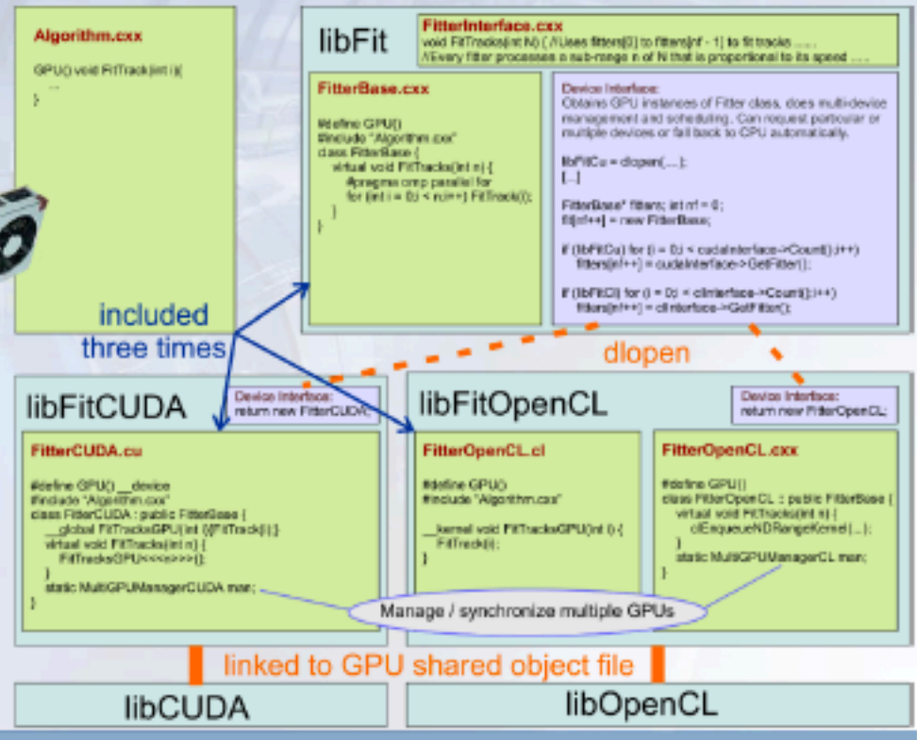
ALICE tracking in Run 3

Compatibility with several GPU frameworks

ALICE

- **Generic common C++ Code compatible to CUDA, OpenCL, HIP, and CPU (with pure C++, OpenMP, or OpenCL).**
  - OpenCL needs clang compiler (ARM or AMD ROCm) or AMD extensions (TPC track finding only on Run 2 GPUs and CPU for testing).
  - Certain worthwhile algorithms have a vectorized code branch for CPU using the Vc library.
  - All GPU code swapped out in dedicated libraries, same software binaries run on GPU-enabled and CPU servers.

- **Screening different platforms for best price / performance.**
  (including some non-competitive platforms for cross-checks and validation.)
  - **CPUs (AMD Zen, Intel Skylake)**
    C++ backend with OpenMP, AMD OCL
  - **AMD GPUs**
    (S9000 with OpenCL 1.2, MI50 / Radeon 7 / Navi with HIP / OCL 2.x)
  - **NVIDIA GPUs**
    (RTX 2080 / RTX 2080 Ti / Tesla T4 with CUDA)
  - **ARM Mali GPU** with OCL 2.x
    (Tested on dev-board with Mali G52)



- **Optimize TCO (faster GPUs → less latency → smaller buffers).**

5.11.2019 David Rohr, drohr@cern.ch 7 / 9

https://indico.cern.ch/event/773049/contributions/3581368

ALICE tracking in Run 3

Data compression

ALICE

- **Data compression mandatory to store minimum bias data.**
  - TPC most critical as largest data contributor (lossy + lossless compression).
    - **Online cluster finding** (basis for entropy compression, entropy coding of raw data insufficient).
    - All cluster properties stored in **individually optimized fixed / floating point format**.
    - Coordinates of unattached clusters sorted, and stored as **differences**.
    - Entropy-compressed via **ANS** (reaching ideal entropy).
    - **Correlated properties** encoded together.
    - Coordinates of clusters attached to tracks stored as **residual to extrapolated track**.
    - Clusters of tracks not used for physics are **removed**.
  - **Strategy B** already yields **sufficient data reduction**, still **working on strategy A**.
  - Other detectors also use **ANS** coding.
    - Optionally use additional steps: **zero suppression**, **cluster shape hashing** (ITS), …
    - Less critical than TPC
  - **Total data rate estimate with strategy B: 71.7 – 89.9 GB/s (TDR: 88 GB/s)**

For track model: see CTD2018 talk: https://indico.cern.ch/event/742783/contributions/3274344/

Tracking needed for data compression

TPC data rejection alternatives

A. Reject only clusters of identified background / tracks (loopers).
Rejects: 12.5% - 39.1%

B. Keep only clusters attached or in proximity of identified signal tracks.
Rejects: 37.3% - 52.5%

For more details, see talk by Ruben Shahoyan
ALICE continuous readout and data reduction strategy for Run 3

For ANS: see poster of Michael Lettrich: Fast and Efficient Entropy Compression of ALICE Data using ANS Coding

5.11.2019 David Rohr, drohr@cern.ch 8 / 9

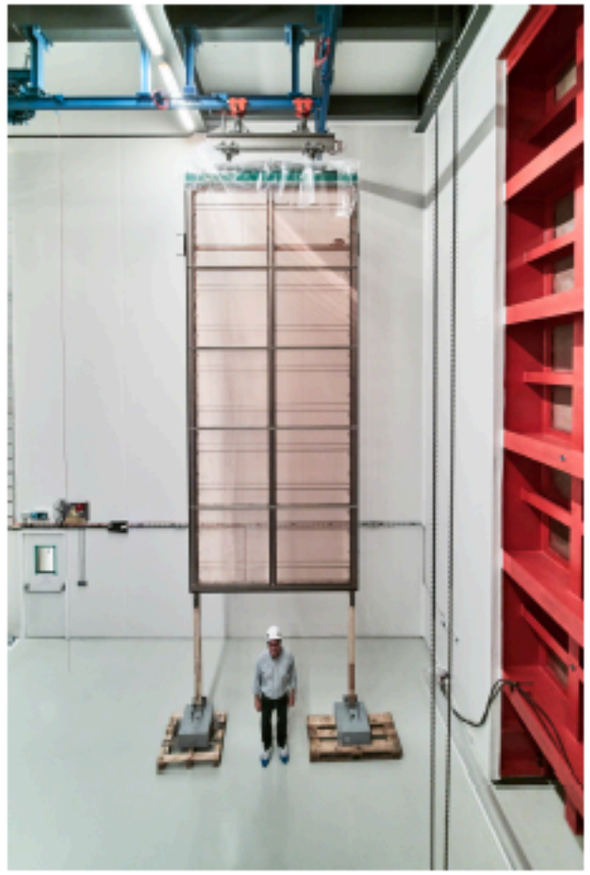
https://indico.cern.ch/event/773049/contributions/3581368

DUNE

LAr TPC data volumes

- The first far detector module will consist of 150 **Anode Plane Assemblies (APAs)** which have 3 planes of wires with 0.5 cm spacing. Total of **2,560** wires per APA
- Each wire is read out by 12-bit ADC's every 0.5 microsecond for 3-6 msec. Total of **6-12k** samples/wire/readout.
- Around 40 MB/readout/APA uncompressed with overheads → **6 GB/module/readout**
- 15-20 MB compressed/APA → **2-3 GB/module/readout**
- Read it out ~5,000 times/day for cosmic rays/calibration → **3-4PB/year/module (compressed)**

(x 4 modules x stuff happens x decade) = ….



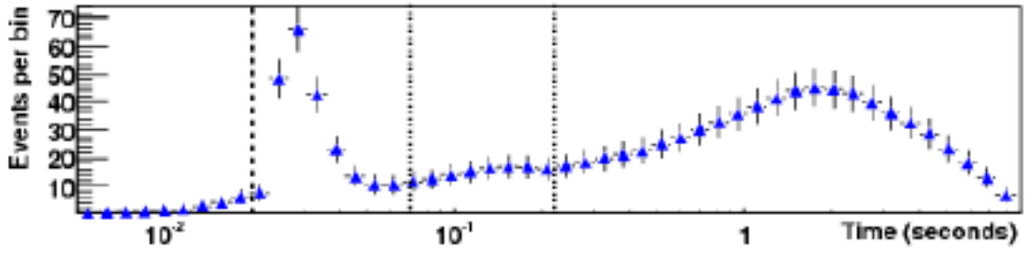
1 APA – 2,560 channels
150 of these per FD module

https://indico.cern.ch/event/773049/contributions/3581360

DUNE

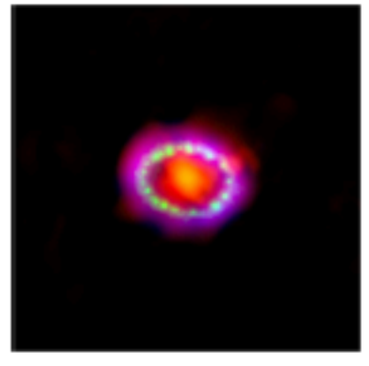
To make it more interesting

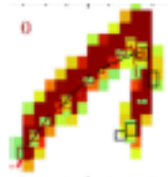
- DUNE should be sensitive to nearby (Milky Way and friends) supernovae. Real ones are every 30-200 years but we expect 1 false alarm/month



- Supernova readout = 100 sec, one trigger/month
- 100 sec readout implies
  - 1 channel = 300 MB uncompressed
  - 1 APA = 768 GB uncompressed
  - 1 module = 115 TB uncompressed
  - 4 SP modules = **460 TB** … takes 10 hrs to read at 100 Gb/s
  - Dual Phase technology has higher S/N → smaller per module
- Some calibration runs will be similar in scope….

Supernova 1987A





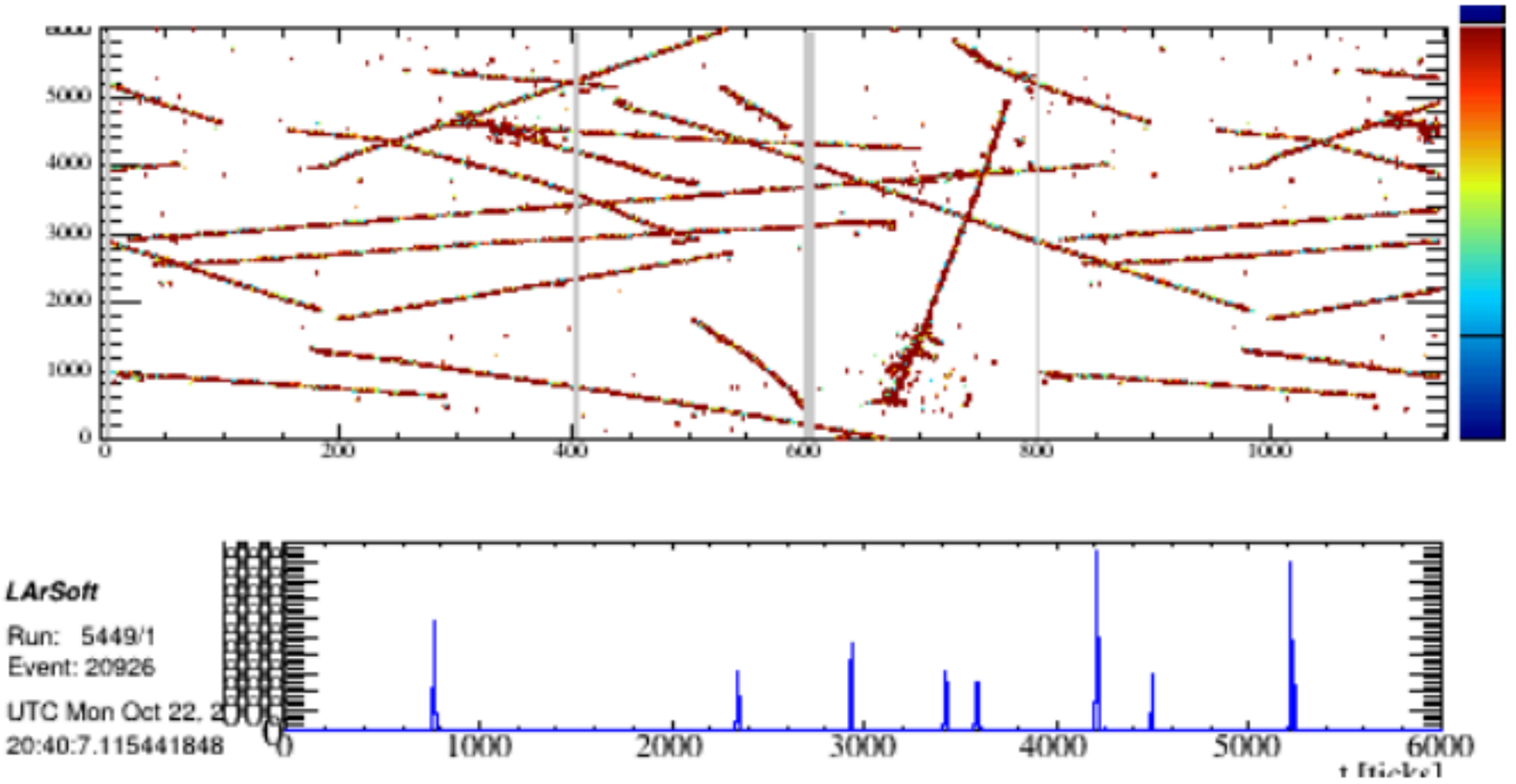
30 MeV ν_eCC

10 MeV NC
$\nu+A \rightarrow \nu+A^*$

https://indico.cern.ch/event/773049/contributions/3581360

DUNE

Signal processing for 1 APA



Remove bad hits, coherent noise, deconvolute, 2560x6000 12 bit

https://indico.cern.ch/event/773049/contributions/3581360

DUNE

Scaling https://indico.cern.ch/event/773049/contributions/3581360

2018: ProtoDUNE event
6 APA ~ 130 MB
At 25 Hz

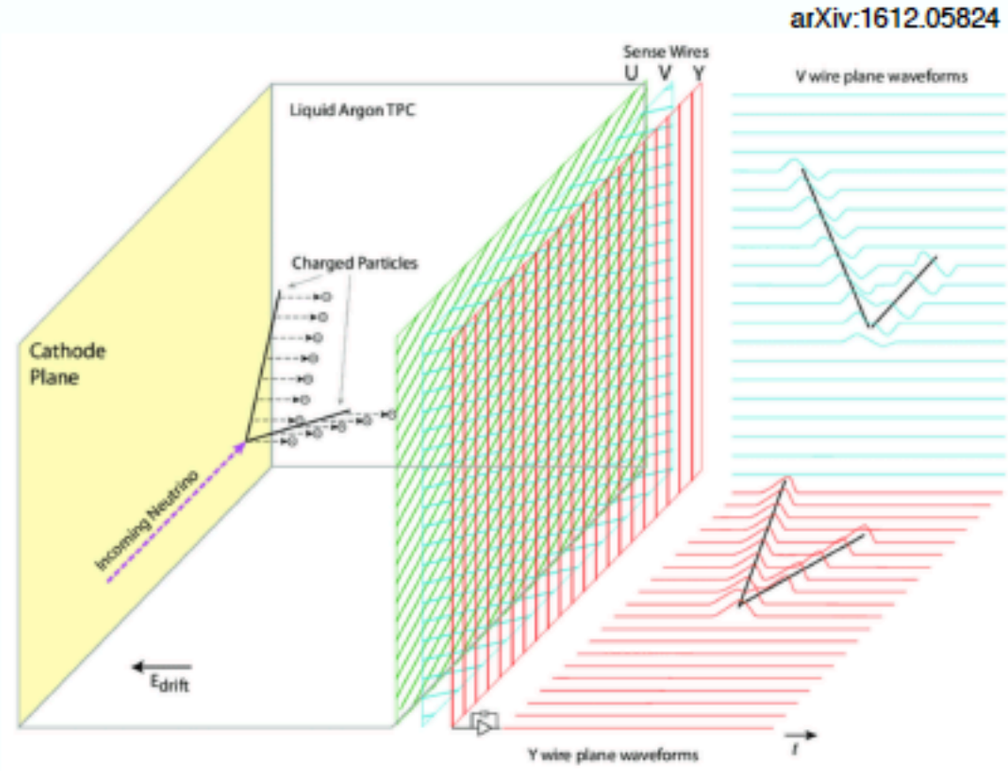
2025: Beam/cosmic ray event
in 1 FD module -- 150 APA ~ 6GB at < 0.1 Hz

Someday: Supernova
150x4x20,000 5 ms APA
~400 TB. 1/month

LArTPC reconstruction

LArTPC Working Principle

- Charged particles produced in neutrino interactions ionize the argon
- Ionization electrons drift in electric field towards anode planes
  - typically 2 induction planes, one collection plane
- Sense wires detect the incoming charge
  - waveforms have bipolar signal on induction planes, unipolar on collection plane
  - wires are oriented at different angles in the planes for 3D reconstruction

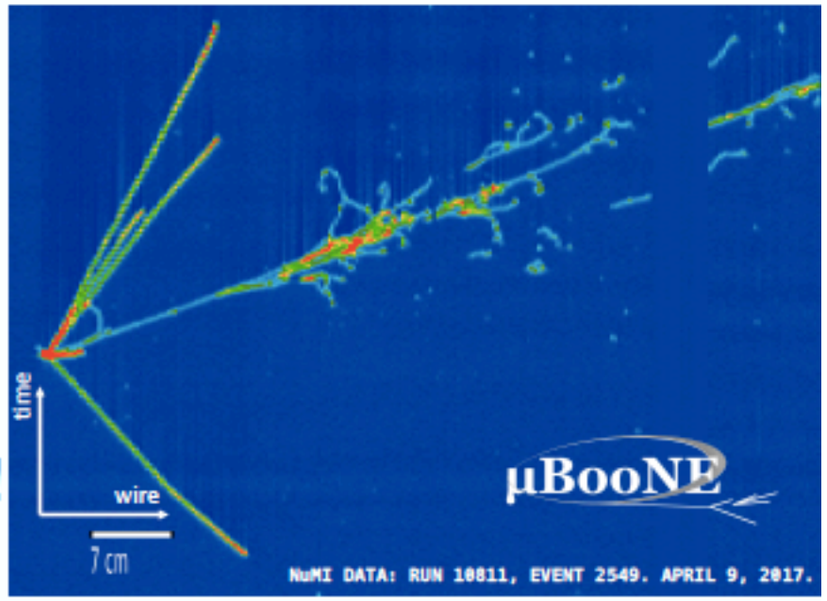


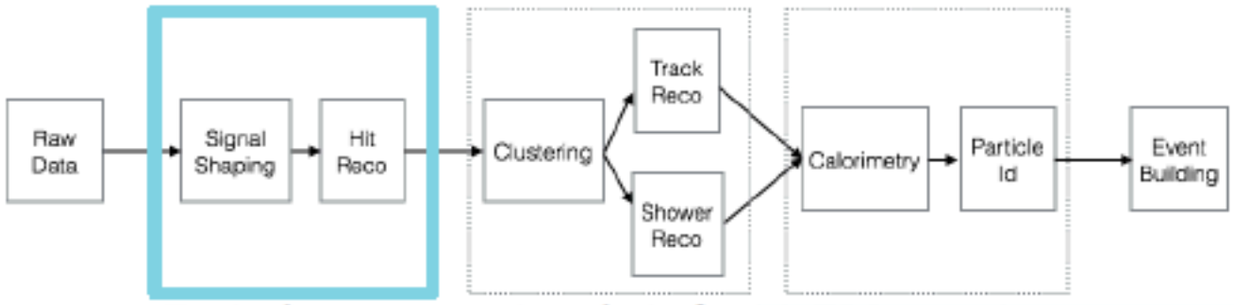
https://indico.cern.ch/event/773049/contributions/3474764

LArTPC reconstruction parallelized

LArTPC Reconstruction

- Reco in LArTPC experiments is **challenging** due to unknown interaction point, many possible topologies, noise, contamination of cosmic rays
  - Takes **O(minutes)/event** in MicroBooNE
  - ICARUS ~5x bigger, DUNE Far Detector O(100)x bigger
- LArTPC detectors are modular in nature → **parallelism!**
- LArTPC is a 'young' technology:
  reconstruction is a field of active development
  - traditional 'hit-based' reconstruction approach
  - image-based reconstruction: DL techniques or tomographic approach
  - early parts of the reco chain are more mature and common between different approaches





Typical reconstruction chain for LArTPC experiments

5 2019/11/07 LArTPC reco - CHEP19

Fermilab

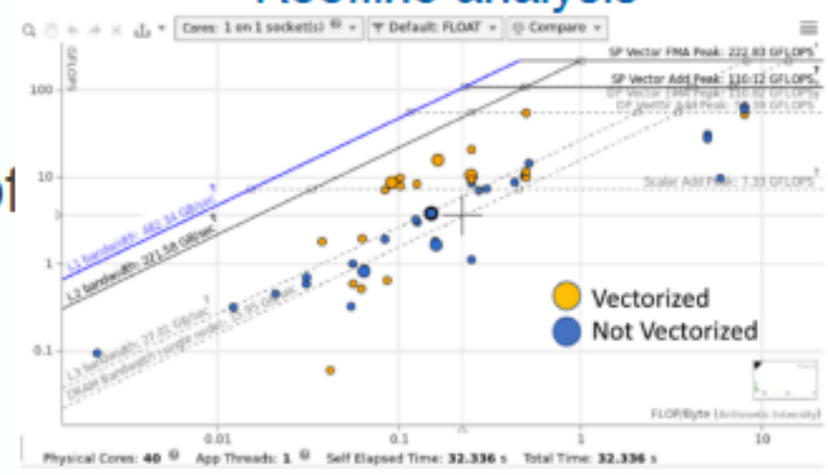
https://indico.cern.ch/event/773049/contributions/3474764

LArTPC reconstruction parallelized

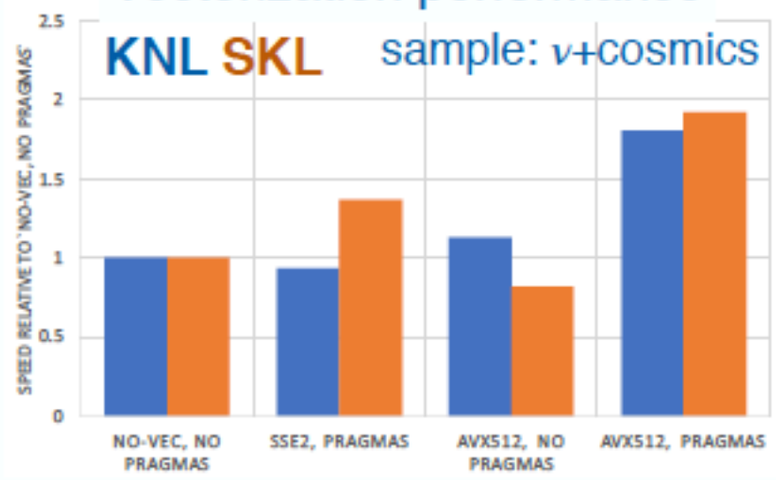
Vectorization Results

- Profiling the code (e.g. roofline) shows that most of the time is still spent in the minimization algorithm
  - number of iterations needed to converge is variable: difficult to vectorize across multiple hit candidates.
- We choose to **vectorize specific loops** within the algorithm, typically **across data bins**
  - main limitations: only a subset of the code is vectorized, number of bins is same order as vector unit size
- Close to **2x speedups**, both on Skylake Gold (SKL) and KNL when compiling with icc+AVX-512
  - Compared to Intel Math Kernel Library: our fitter is faster

Roofline analysis



Vectorization performance



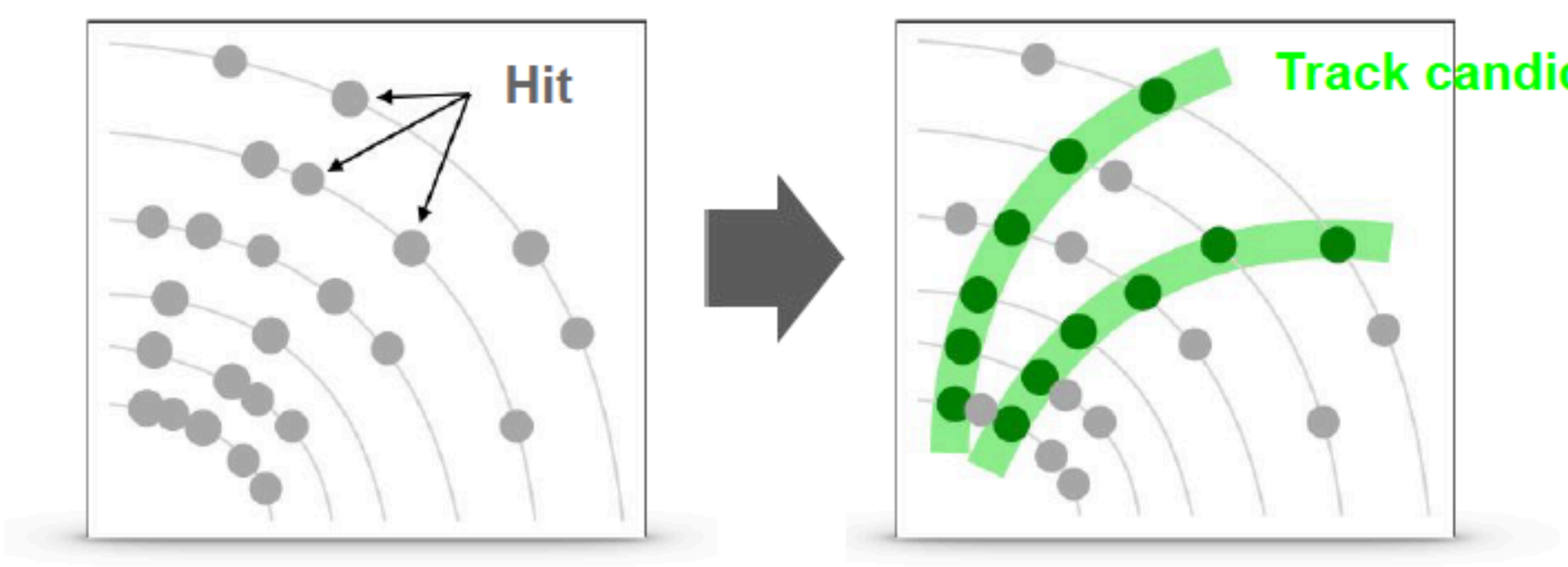
Fermilab

https://indico.cern.ch/event/773049/contributions/3474764

More distant future?: Quantum computing

Track reconstruction

- Find the correct set of hits belonging to the same particle
  - Hits are distributed according to known physics rule: helix curve, scattering with material…
- Combinatorial optimization problem
  - HL-LHC environment (<mu>=200)
    - 10^4 particles, ~10 detector layers. i.e. 10^5 hits
  - Does quantum computing favor such a problem?



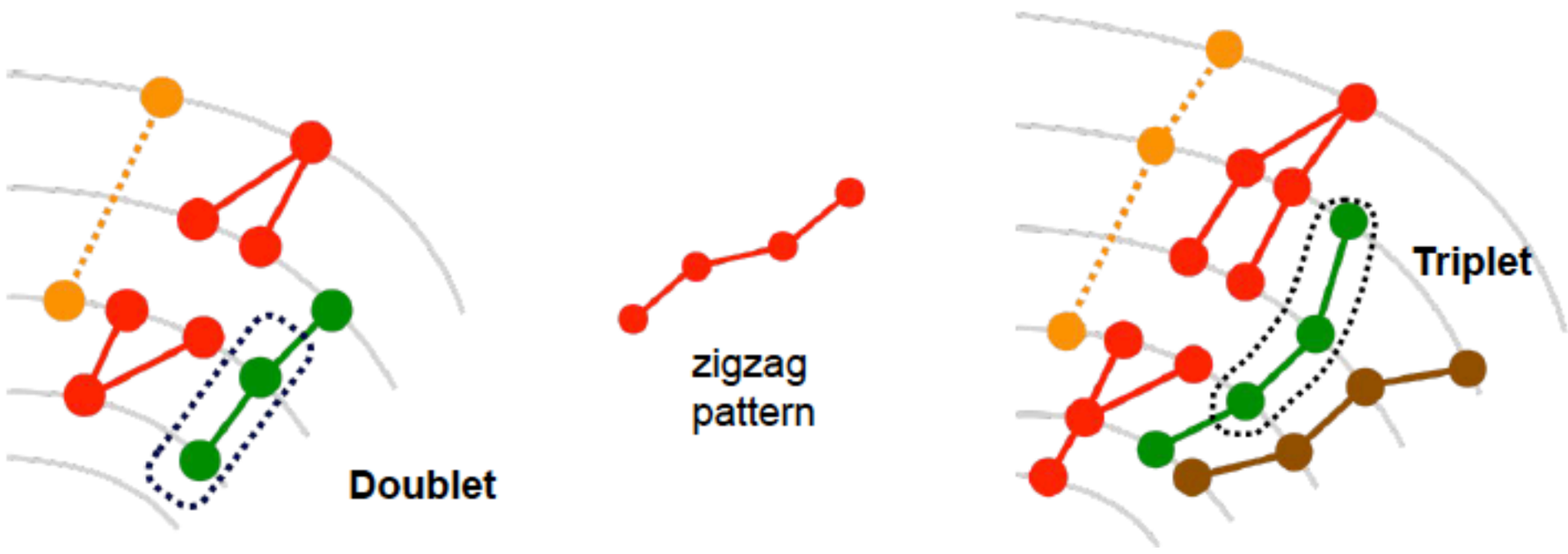
3

https://indico.cern.ch/event/773049/contributions/3474750

More distant future?: Quantum computing

Qubit assignment

- Doublets
  - Original paper (Stimpfl-Abele, et. al., 1991)
  - Zigzag pattern, many qubits required due to many possible fake candidates
- Triplets
  - This work (arxiv: 1902.08324)
  - Strong fake reduction, resulting in a reasonable number of qubits



https://indico.cern.ch/event/773049/contributions/3474750

More distant future?: Quantum computing

Hamiltonian

$$E = \alpha \left(\sum_i^N T_i \right) - \left(\sum_{i,j} S_{ij} T_i T_j \right) + \zeta \left(\sum_{i,j} T_i T_j \right), \; T \in \{0,1\}$$

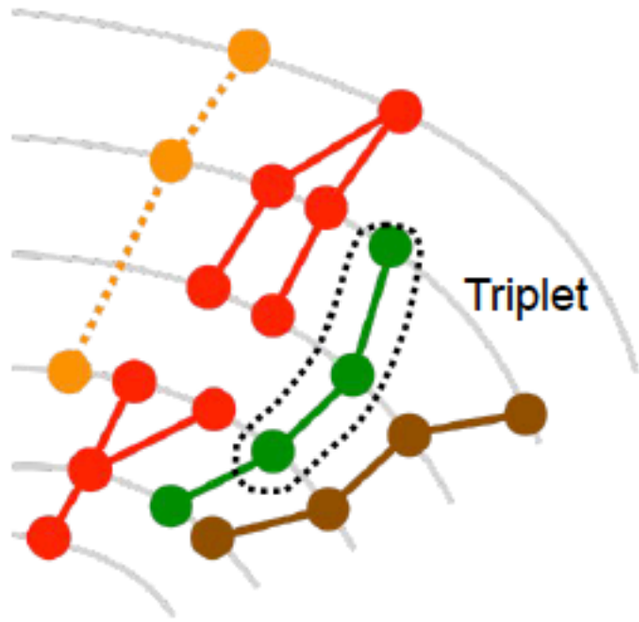
qubit

bias weight Connection strength Avoid conflicts, zigzag pattern, holes

Quadratic Unconstrained Binary Optimization (QUBO)

$$O(a,b,T) = \sum_i^N a_i T_i + \sum_i^N \sum_{j<i}^N b_{ij} T_i T_j$$

$$b_{ij} = \begin{cases} -S(T_i, T_j) & T \in \{0,1\} \\ \zeta \\ 0 \end{cases}$$



7

https://indico.cern.ch/event/773049/contributions/3474750

More distant future?: Quantum computing

D-Wave Quantum Annealing Machine

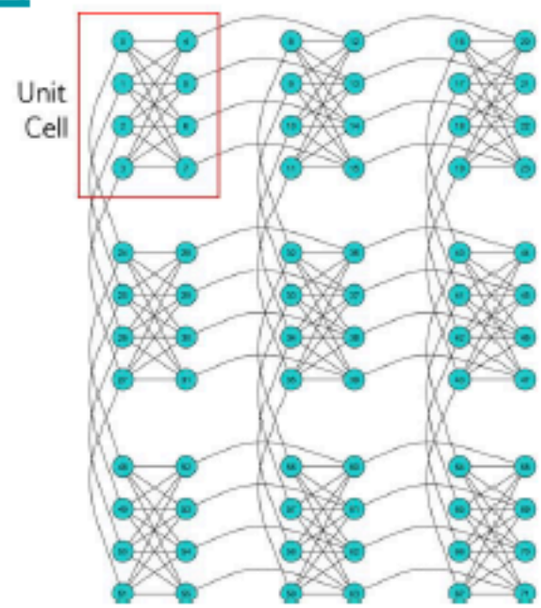
[D-wave 2000Q]

- Superconducting qubits (cool down to 15 mK)
- 2048 qubits (D-wave 2000Q)
- Chimera graph
  - 16x16 units, 8 qubits / unit
    - 6016 couplers
  - ~ 64-bit full connection
- Annealing time 1-2000 μs

Next-generation machine:

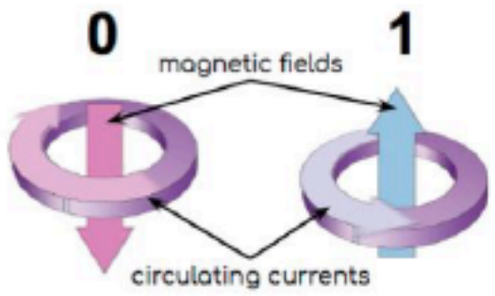
- Pegasus processor
- 5000 qubits
- 2020 mid

Chimera graph

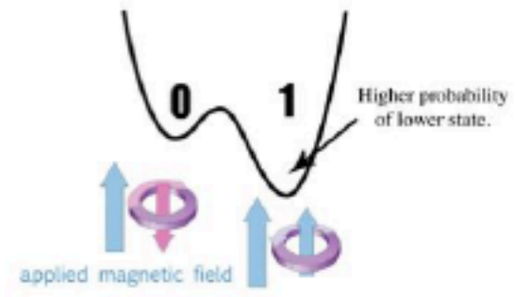


$$O(a, b, T) = \sum_{i}^{N} a_i T_i + \sum_{i}^{N} \sum_{j<i}^{N} b_{ij} T_i T_j$$

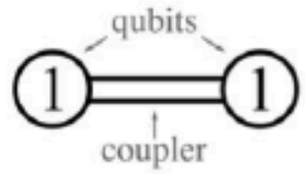
qubit: T_i



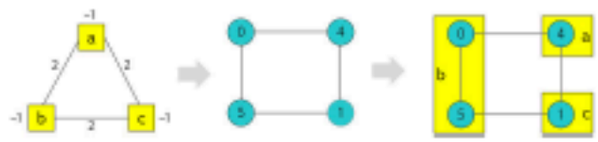
bias weight: a_i



coupling strength: b_{ij}



Minor embedding



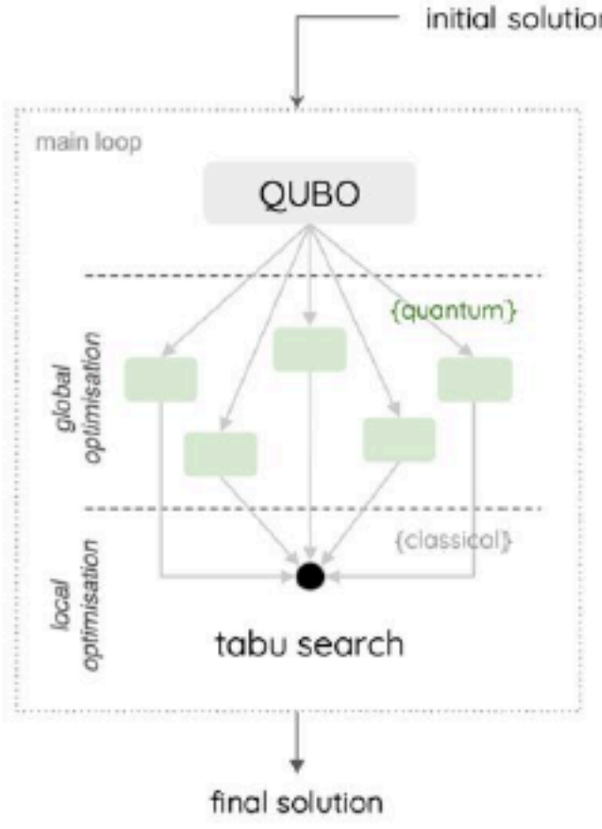
9

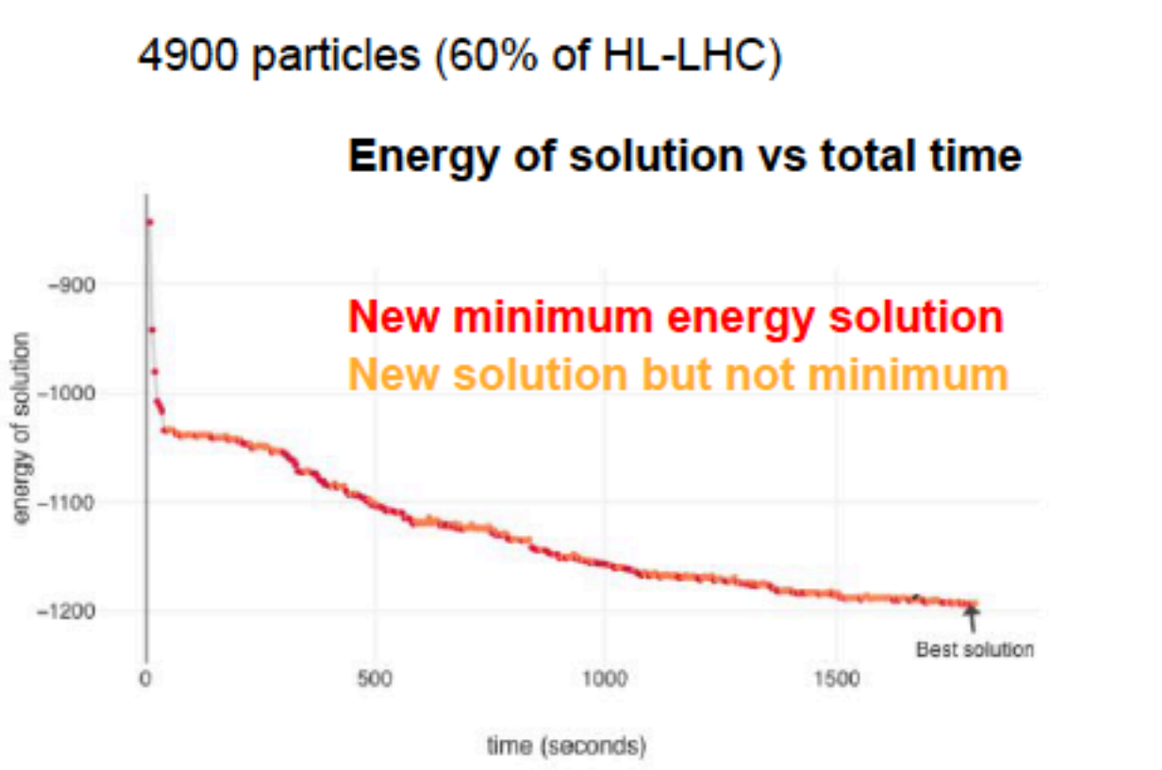
https://indico.cern.ch/event/773049/contributions/3474750

More distant future?: Quantum computing

Solving

- Large QUBOs split into sub-QUBOs due to limited number of qubits
  - Solve each small QUBOs using D-wave hardware
- Repeat annealing to guarantee an optimal solution





Construct track candidates using doublets in final triplets

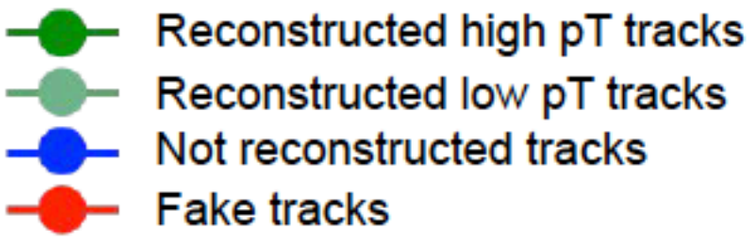
10

https://indico.cern.ch/event/773049/contributions/3474750

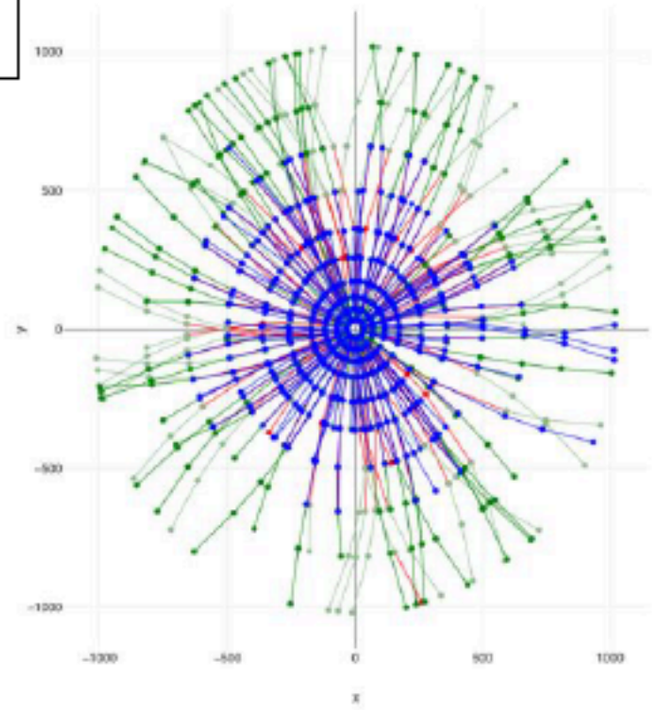
More distant future?: Quantum computing

Results

1600 particles (20% of HL-LHC)
- 11000 hits



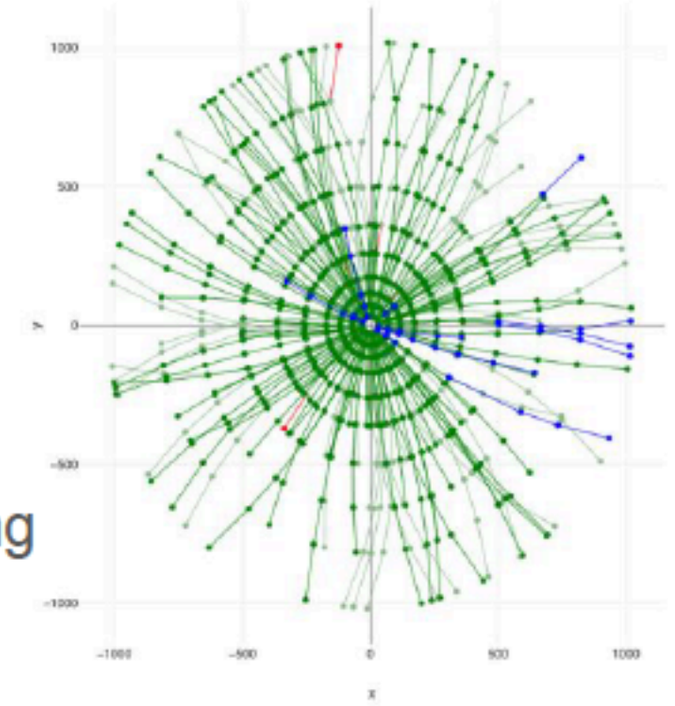
Input



Doublet selection

390000 Doublets	
Purity	0.22 %
Efficiency	99.5 %

Annealing



2445 Doublets

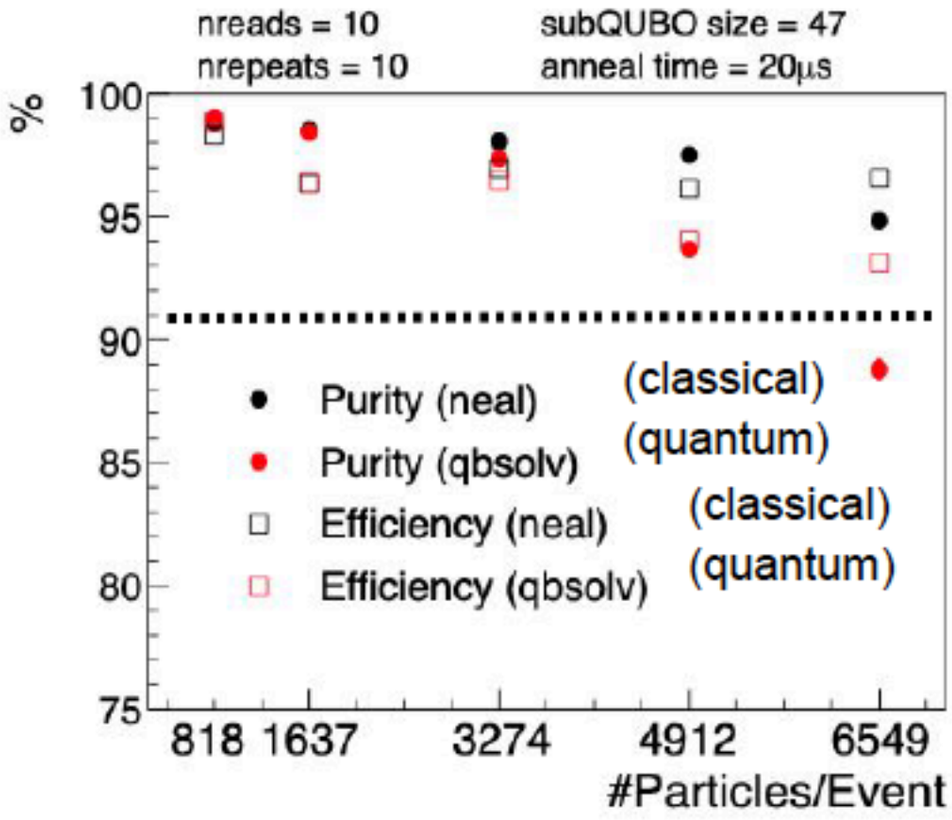
1424 Doublets	
Purity	98.5 %
Efficiency	96.4 %

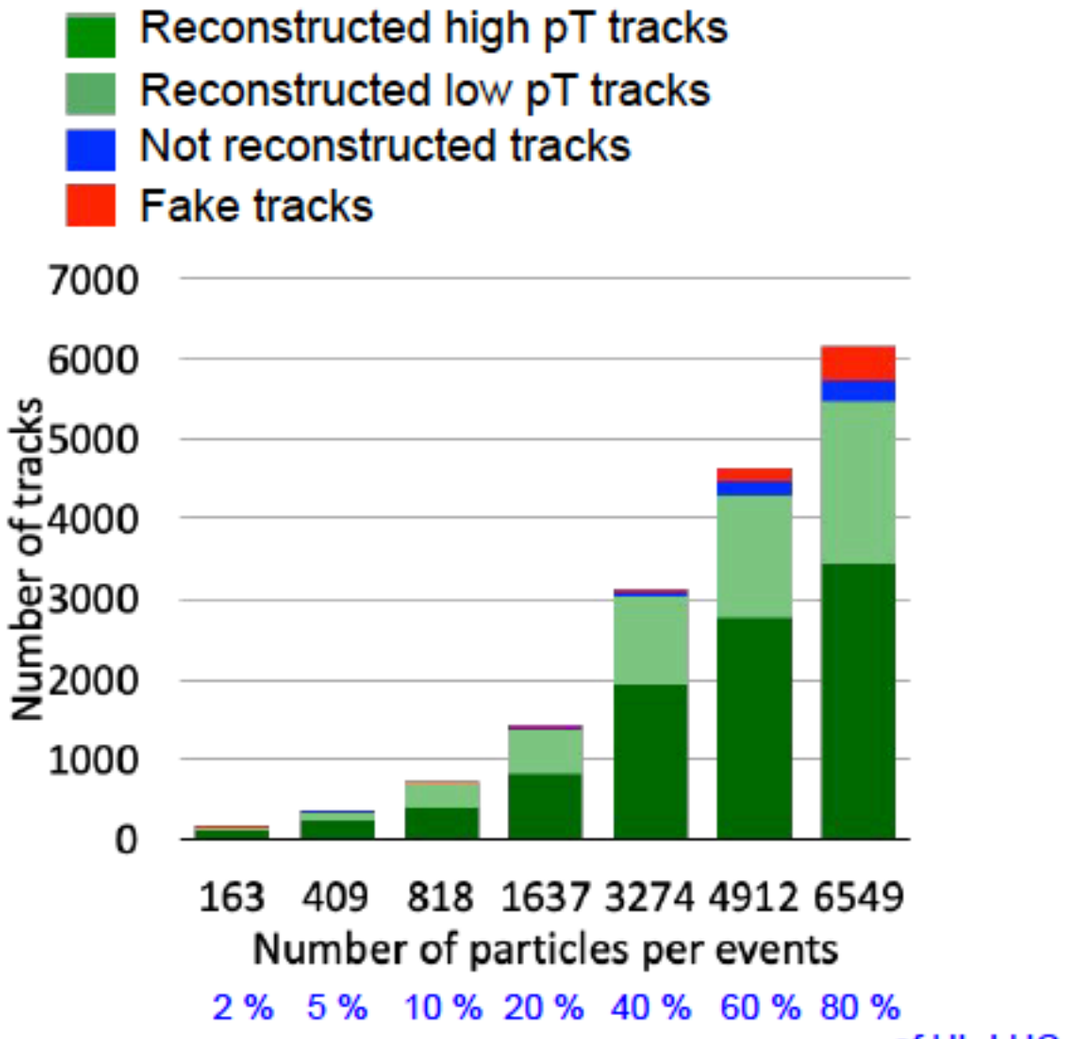
11

We simplify the dataset by focusing on the barrel region of the detector, i.e. hits in the end-caps are removed. If a particle makes multiple energy deposits in a single layer, all but one energy deposits are removed. Hits from particles with $p_T < 1$ GeV and particles with less than five hits are kept and thus part of the pattern recognition, but are not taken into account when computing the performance

More distant future?: Quantum computing

Results





- Reference solver: neal = simulated annealing using CPU
- >90 % efficiency / purity below 6000 particles environment
- Equivalent performance with the classical annealing (neal)

12

https://indico.cern.ch/event/773049/contributions/3474750

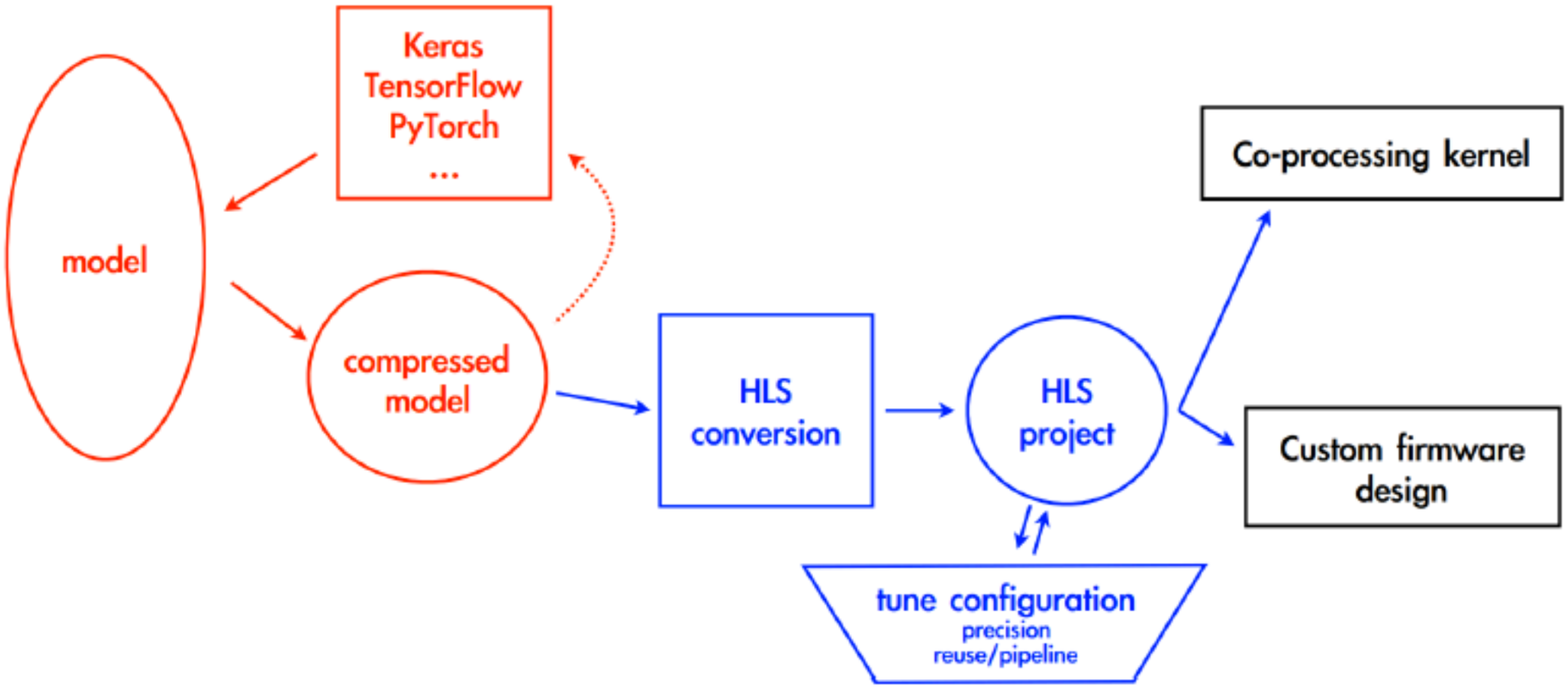
ML on FPGAs

- ***hls4ml*** is a software package for creating HLS implementations of neural networks
  - https://hls-fpga-machine-learning.github.io/hls4ml/
- Supports common layer architectures and model software
- Highly customizable output for different latency and size needs
- Simple workflow to allow quick translation to HLS

https://indico.cern.ch/event/797510/contributions/3313676

ML on FPGAs

Design Workflow



- Design model with standard software tools (Keras, Tensorflow, PyTorch)
- Pass network architecture and weights/biases along with configuration parameters to `hls4ml` (creates HLS project)
- Interface HLS code with desired project

17

https://indico.cern.ch/event/797510/contributions/3313676

ML on FPGAs

hls4ml /SDAccel Workshop

- Organized workshop for hls4ml and acceleration last week
  - *How to do ultrafast DNN inference on FPGAs*
  - https://indico.cern.ch/event/769727/
- Lots of interest in across many HEP experiments, industry
  - 90 participants
- By the end of the course all participants were able to actually run inference on FPGAs
  - Used AWS to provide machines for development/acceleration
- Interested in replicating workshop at other locations

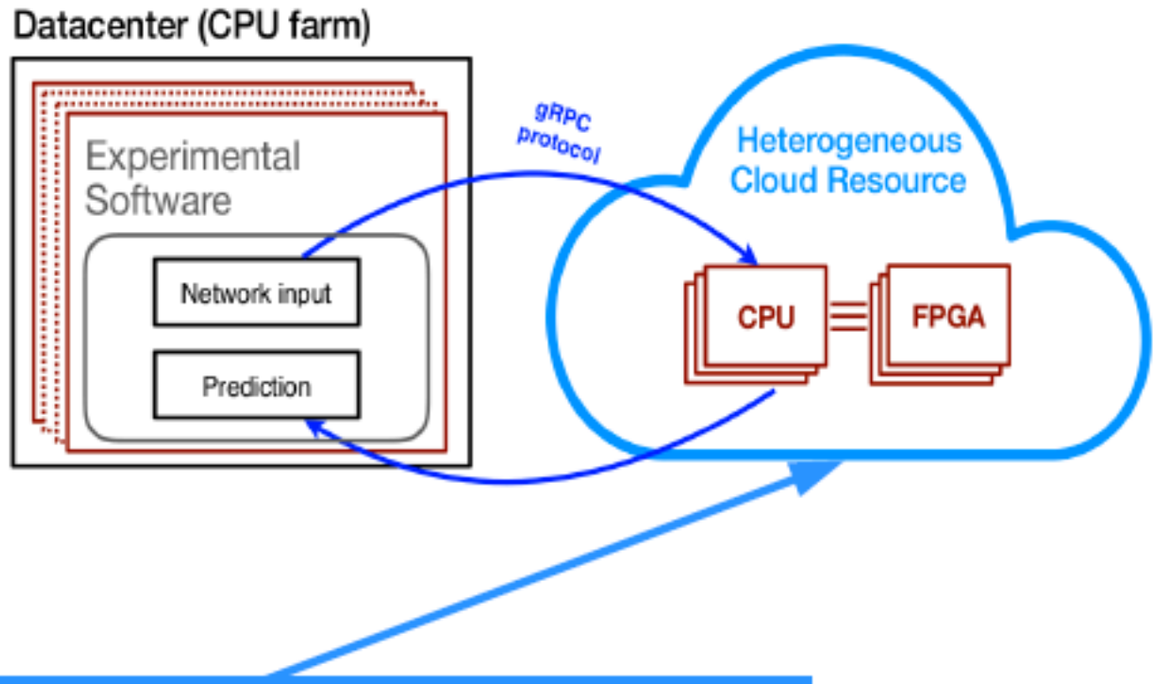
34

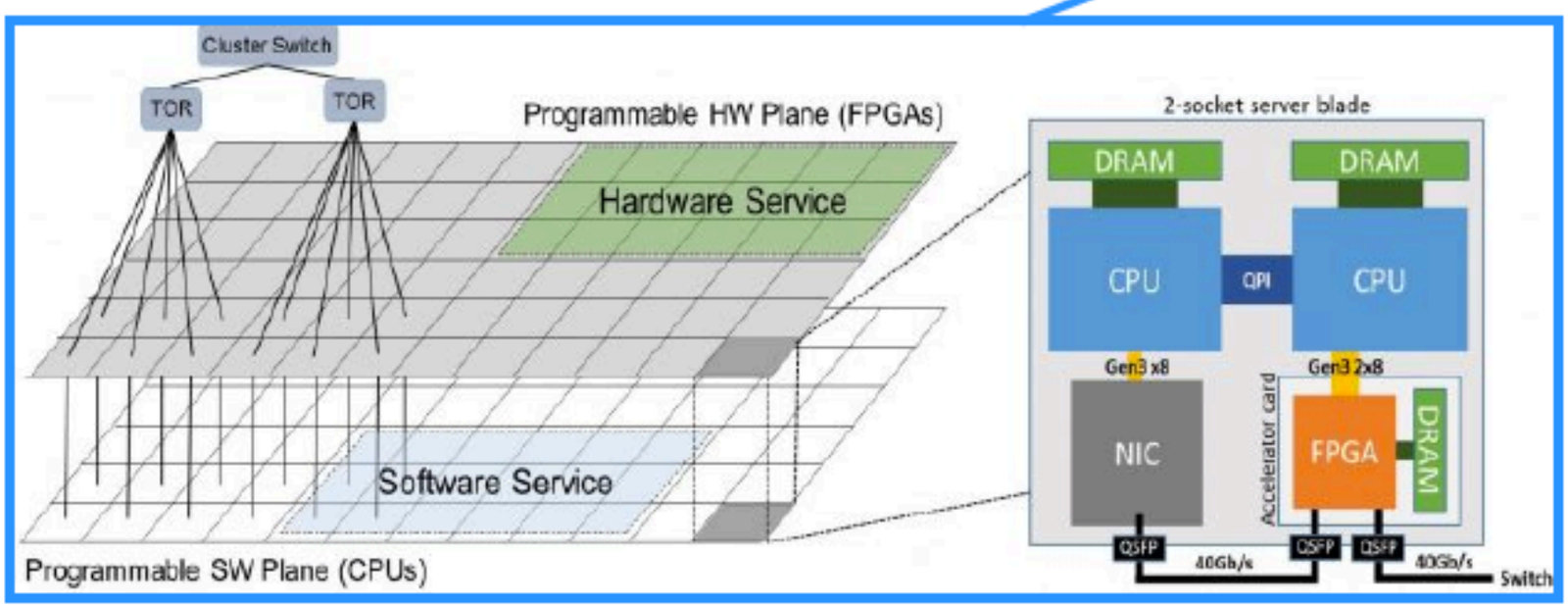
https://indico.cern.ch/event/797510/contributions/3313676

ML on FPGAs

Microsoft Brainwave

- Microsoft Brainwave is a CPU-FPGA server farm
- ML offered “as a service”
  - Send a preprocessed image to Brainwave, get prediction back from ResNet50
- Extremely interesting option for longer latency (>10 ms) systems
- Stay tuned for another talk in the future





36

https://indico.cern.ch/event/797510/contributions/3313676

FPGA inference as a service

SONIC in CMSSW

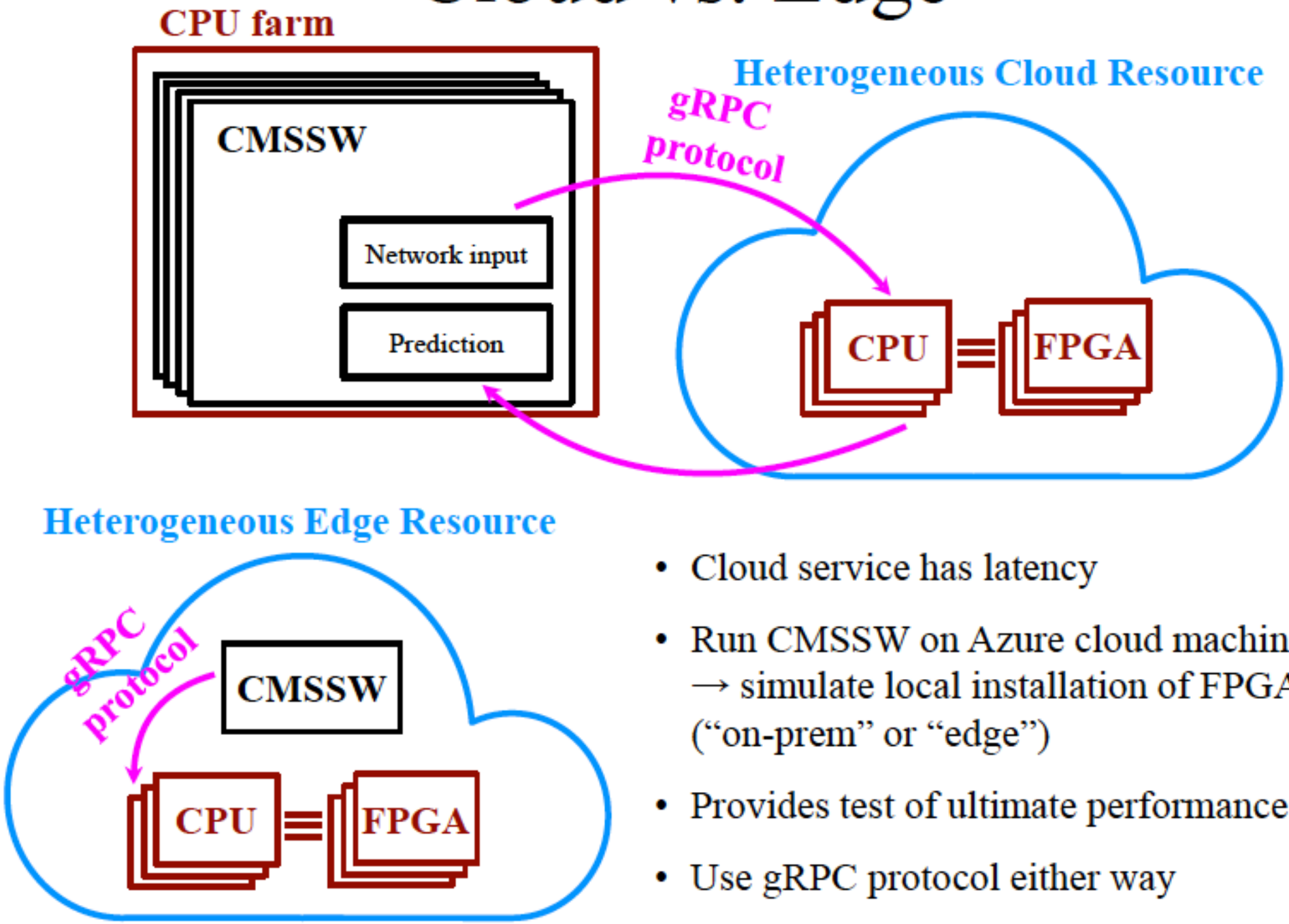
- **S**ervices for **O**ptimized **N**etwork **I**nference on **C**oprocessors
  - Convert experimental data into neural network input
  - Send neural network input to coprocessor using communication protocol
  - Use ExternalWork mechanism for asynchronous requests
- Currently supports:
  - gRPC communication protocol
    - Callback interface for C++ API in development
      → wait for return in lightweight std::thread
  - TensorFlow w/ inputs sent as TensorProto (protobuf)
- Tested w/ Microsoft Brainwave service (cloud FPGAs)
- gRPC SonicCMS repository on GitHub

HOW2019 Kevin Pedro 12

https://indico.cern.ch/event/759388/contributions/3303405 https://arxiv.org/abs/1904.08986

FPGA inference as a service

Cloud vs. Edge



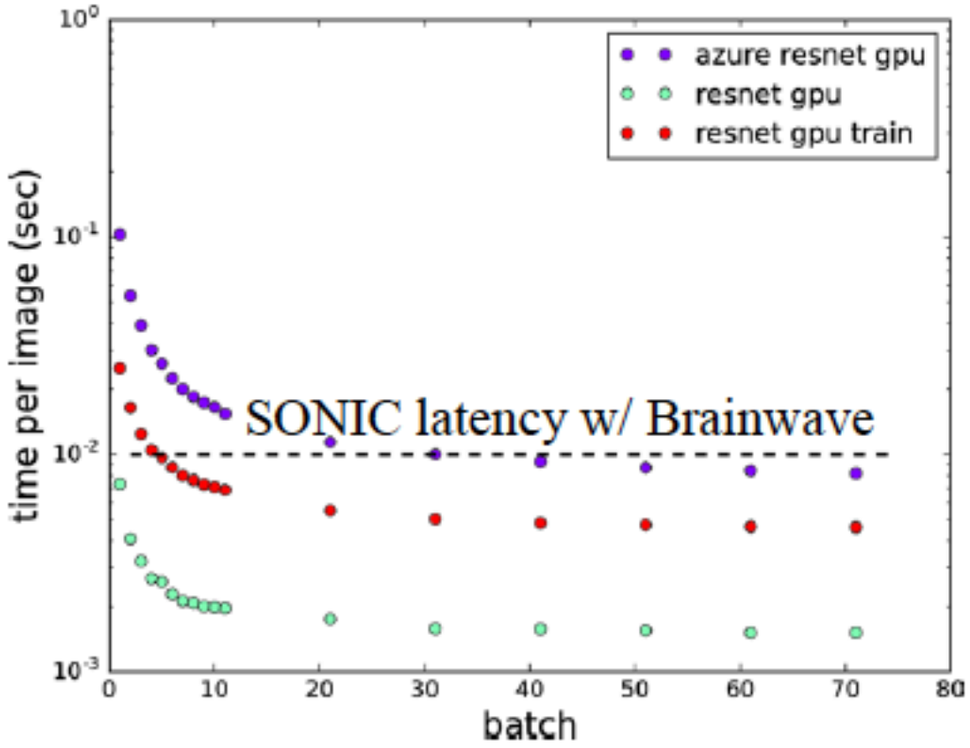
- Cloud service has latency
- Run CMSSW on Azure cloud machine → simulate local installation of FPGAs ("on-prem" or "edge")
- Provides test of ultimate performance
- Use gRPC protocol either way

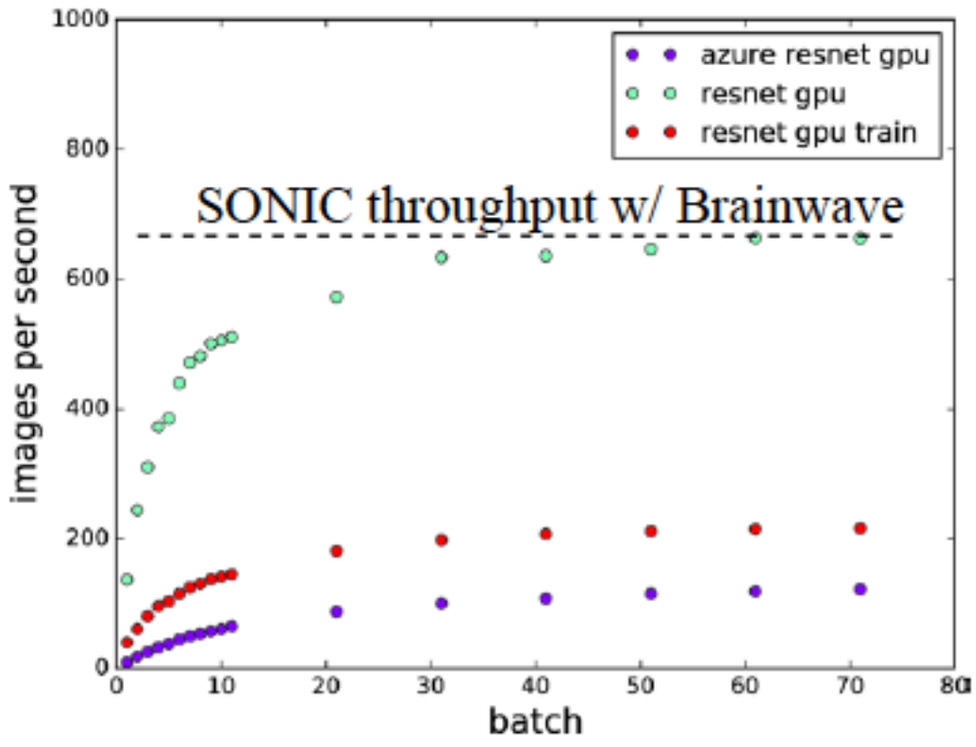
HOW2019 Kevin Pedro 13

https://indico.cern.ch/event/759388/contributions/3303405 https://arxiv.org/abs/1904.08986

FPGA inference as a service

GPU Performance





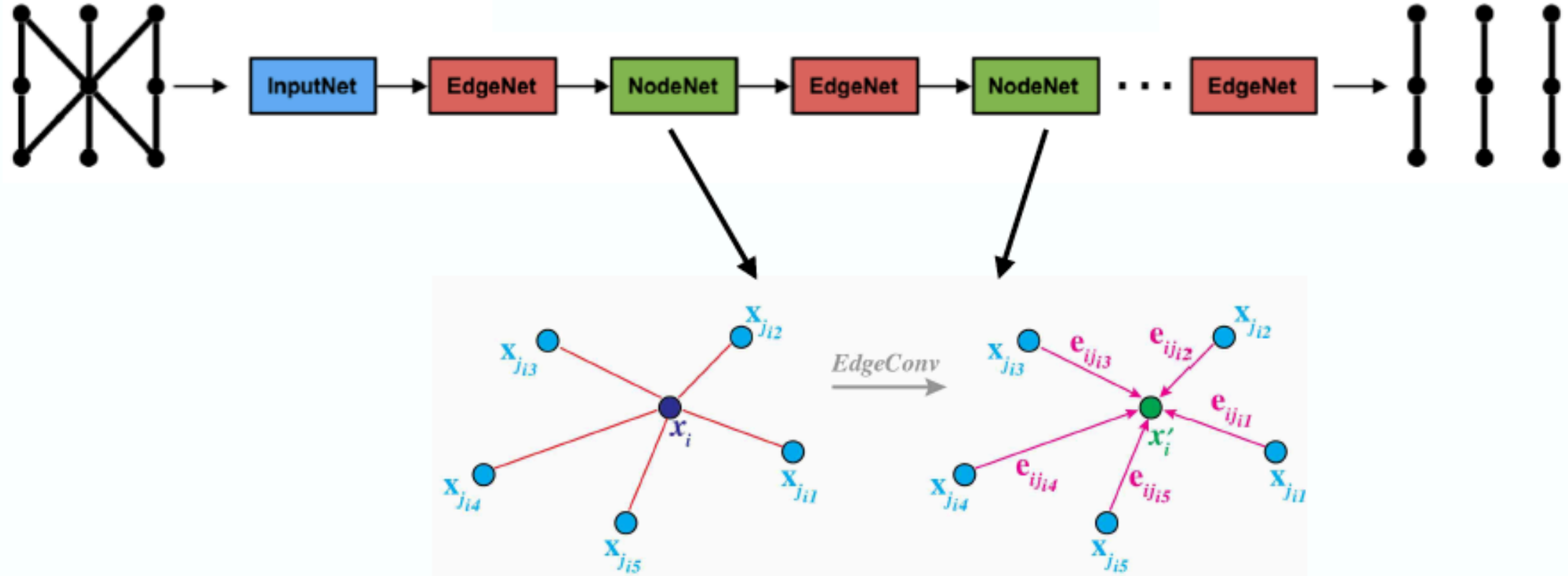
- Above plots use NVidia GTX 1080, TensorFlow v1.10
- GPU directly connected to CPU via PCIe
- TF built-in version of ResNet-50 performs better on GPU than quantized version used in Brainwave

https://indico.cern.ch/event/759388/contributions/3303405 https://arxiv.org/abs/1904.08986

ML graph networks: tracking

A Path Forward with Hep.TrkX

- Aim of project was to discover NN track finders
  - Very promising solution in graph neural networks
  - Particular flavor: graph convolutional edge classifiers



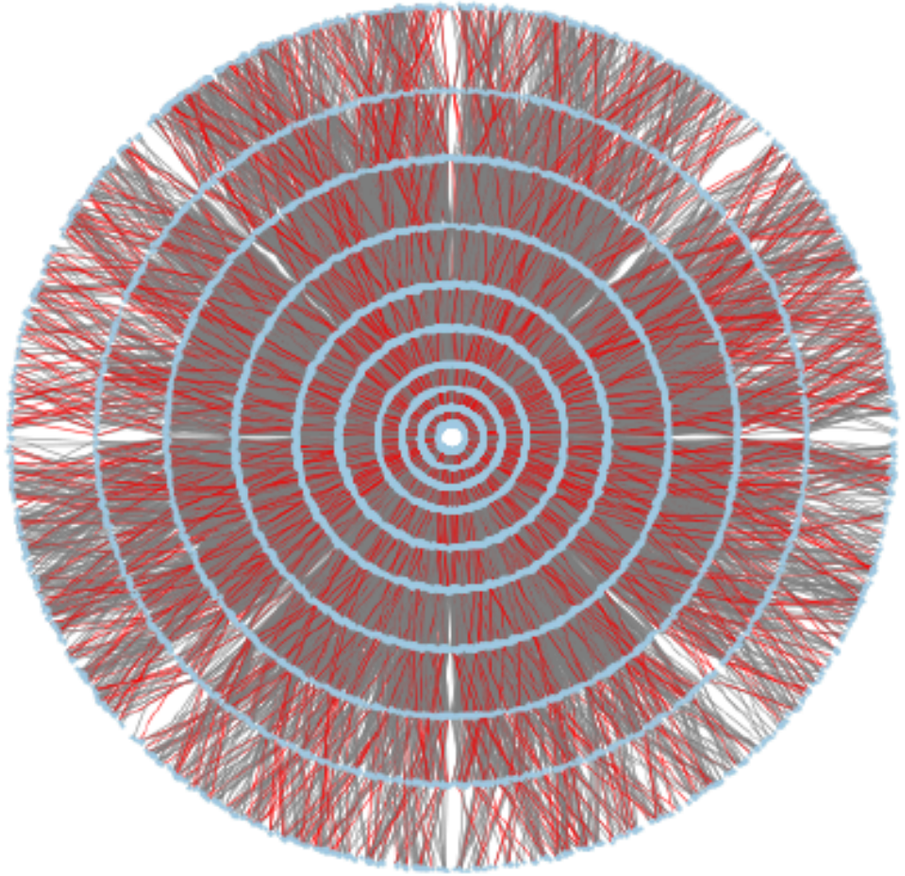
3 https://arxiv.org/abs/1810.06111 Lindsey Gray, FNAL

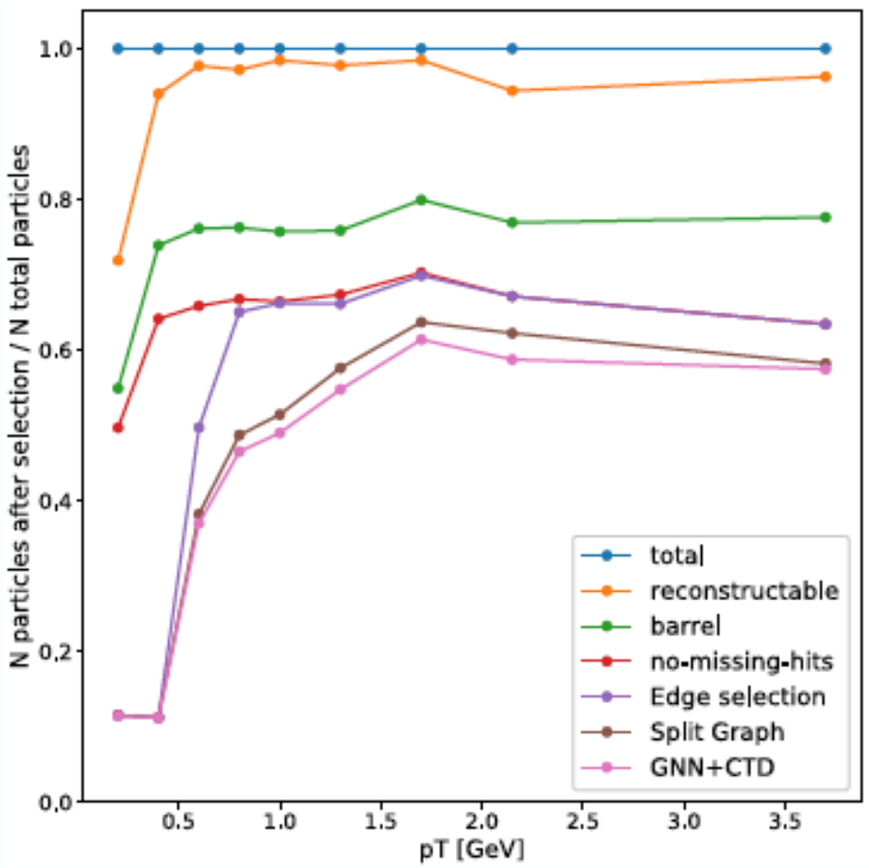
https://indico.cern.ch/event/858670/

ML graph networks: tracking

First Performance and Promise

- Many selections applied to yield training set
  - Important: sectorization and no missing hits
  - These are "easy" tracks but this also early days for the these kinds of network in HEP
- Applying GNN, assembling tracks -> 97% efficient relative to preselection
  - Track-segment selection GNN executes significantly faster than Kalman filter



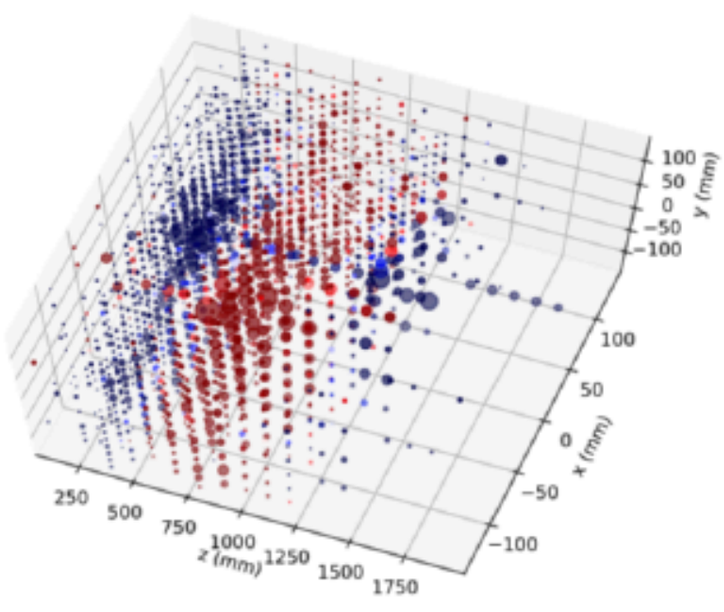


4 X. Ju @ CTD 2019 Lindsey Gray, FNAL https://indico.cern.ch/event/858670/

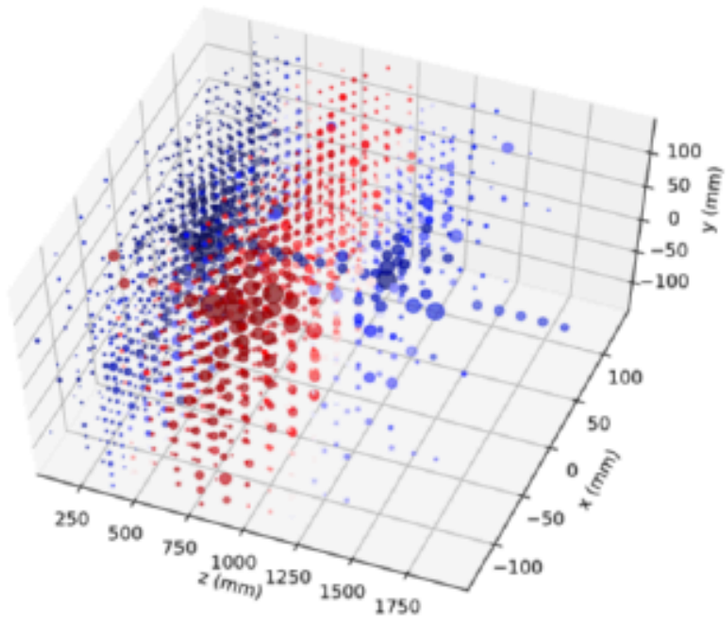
ML graph networks: calorimeter clustering

Finally: Towards Actual Clustering

- Finding correct edges in a static graph is a well characterized learning problem
  - And you can use simple algorithms to keep your data processing quick!
- Clustering is better characterized as "finding the correct graph" rather than finding the correct subgraph
  - It's possible to learn the the latent space which best clusters the data
  - But! Right now, the number of clusters / categories needs to be known beforehand



(a) Truth



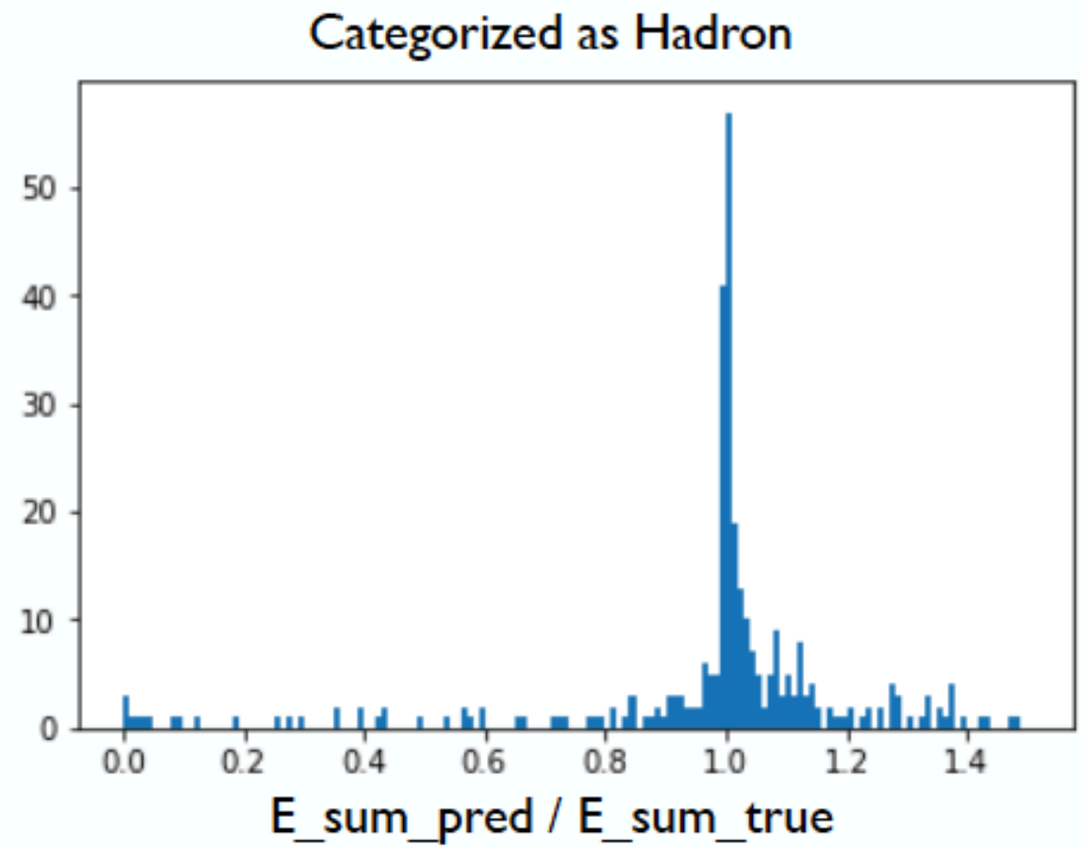
(b) Reconstructed

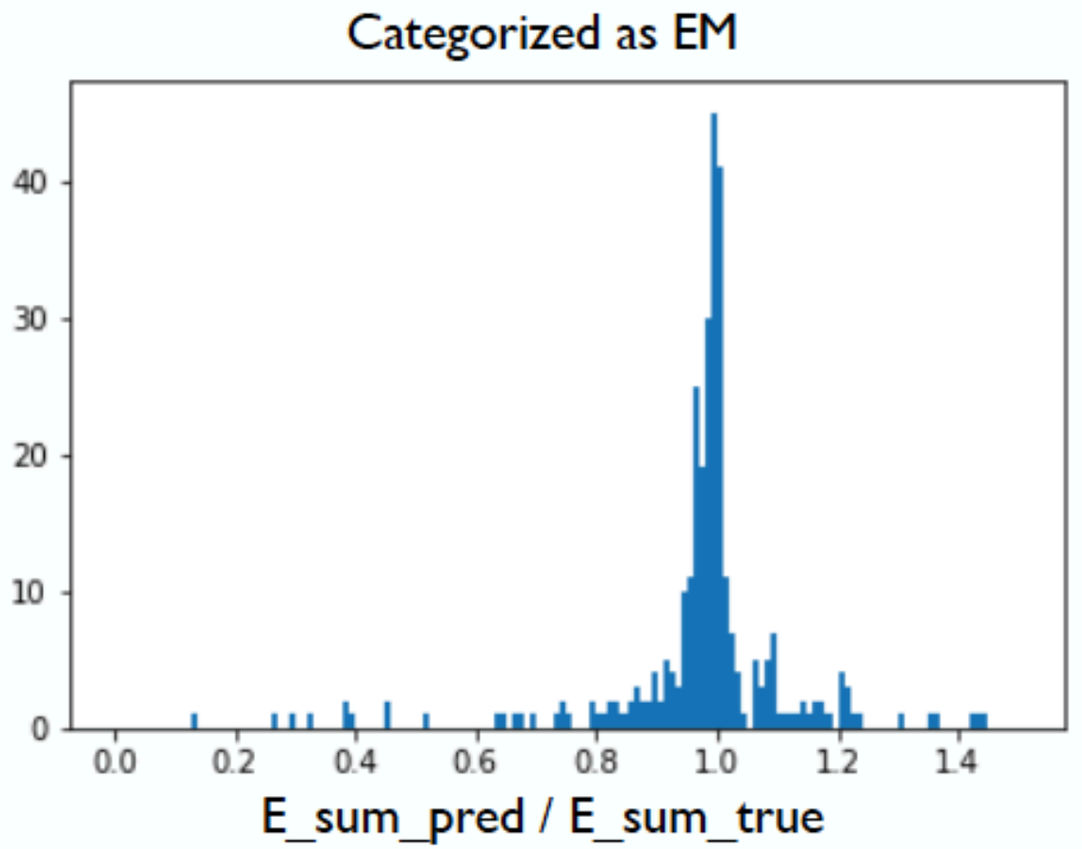
https://arxiv.org/abs/1902.07987

https://arxiv.org/abs/1801.07829

ML graph networks: calorimeter clustering

Energy Categorization





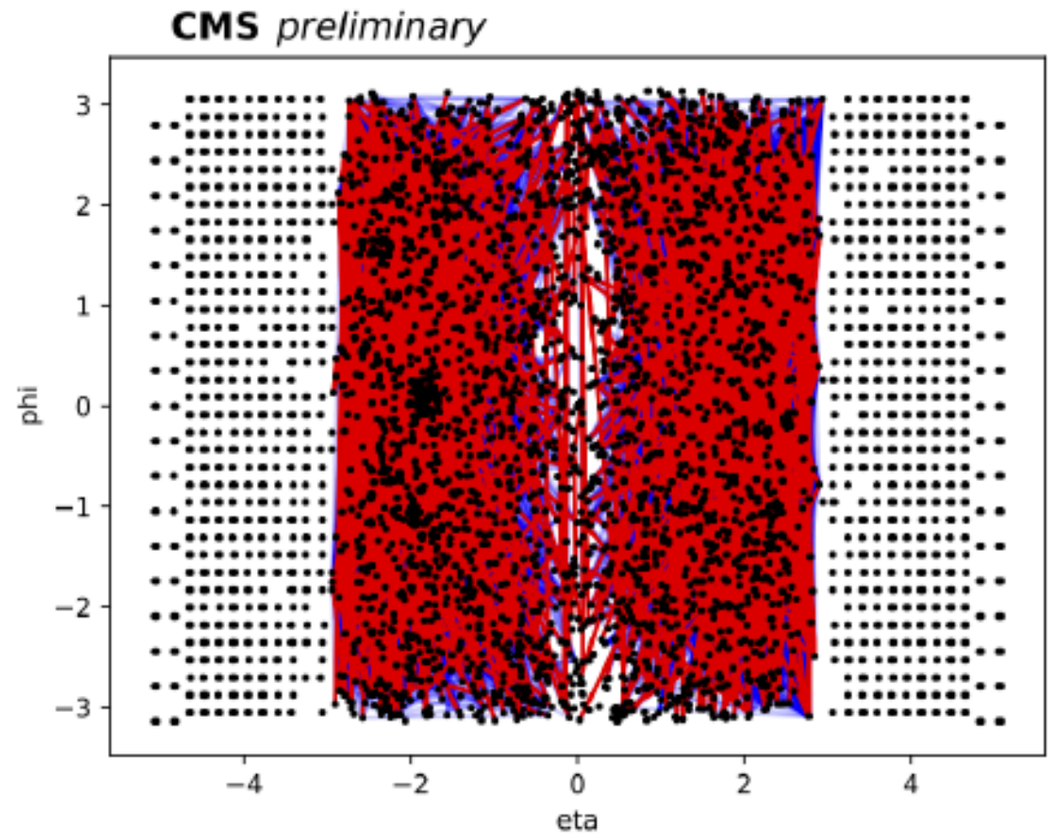
This includes low energy pions and photons, easy to misclassify.
It is important only that these plots are reasonably strongly peaked.

17 S. An / LG / T. Klijnsma / K. Pedro Lindsey Gray, FNAL https://indico.cern.ch/event/858670/

ML graph networks: down to particles

Recent Developments: Particle Flow

- Particle flow is yet another clustering problem
  - Cluster tracks, calorimeter energy deposits, other detector information into things that behave conceptually like gem-particles
- First explorations here are using edge classifiers
  - Other architectures to be considered



Work in progress: ~80% edge efficiency first try