The challenge of Heterogeneous Computing: the CMS case for 2020s

Felice Pantaleo
Experimental Physics Department – CERN
felice@cern.ch

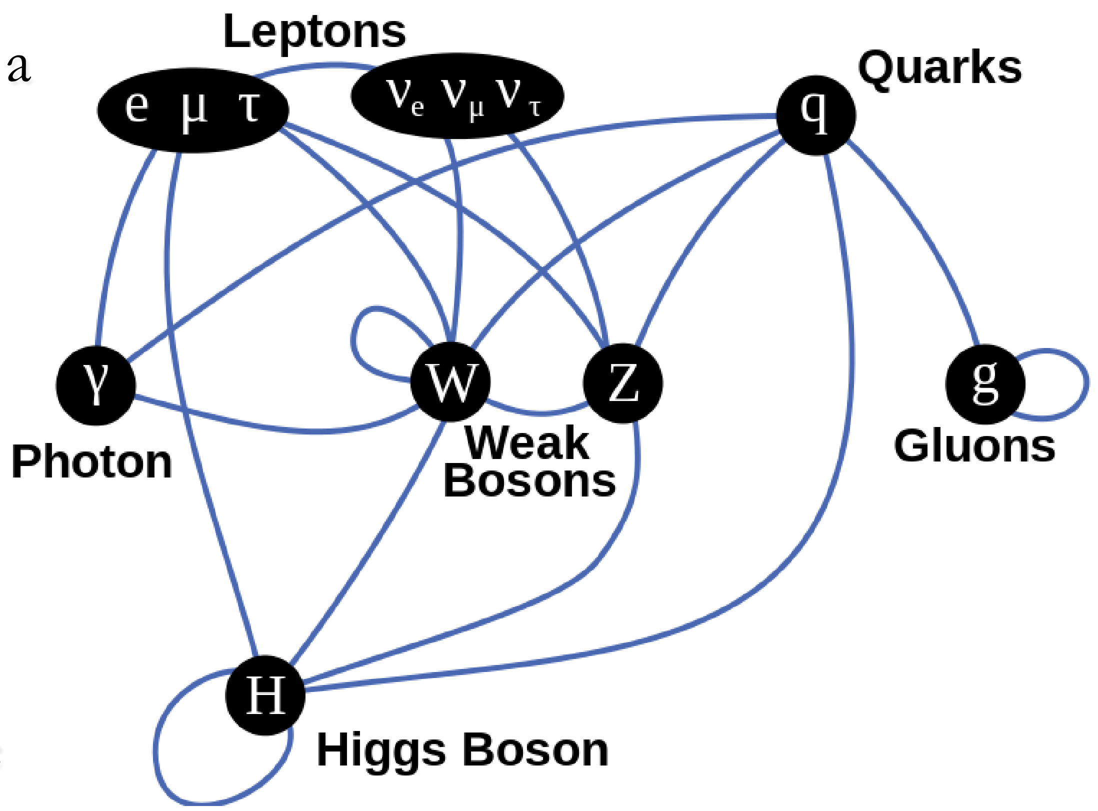
Standard Model

In the context of particle physics, the interactions between fundamental constituents are described by a single theory, the **Standard Model (SM)**,

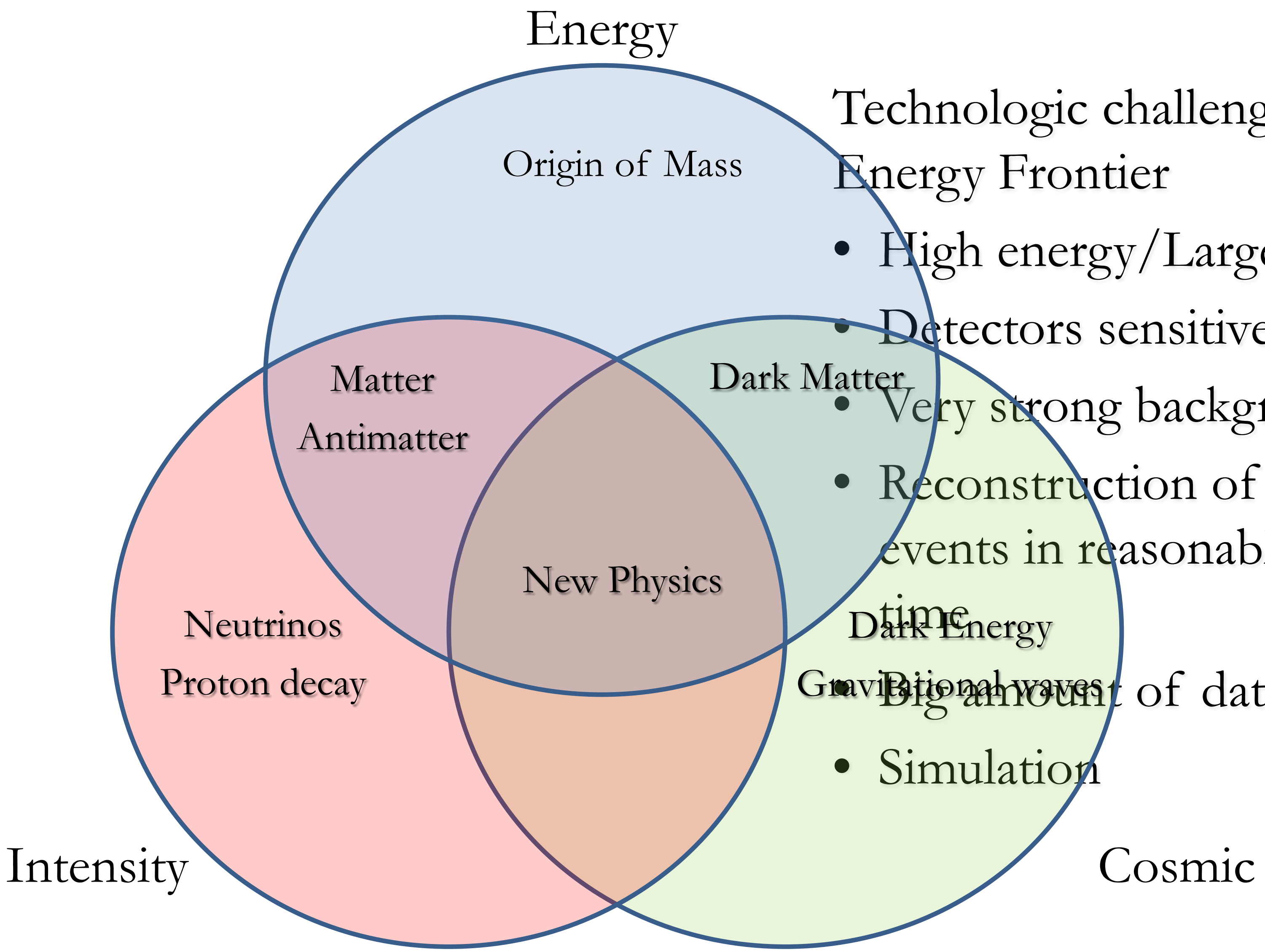
– strengthened by the discovery of the **Higgs boson.**

Aspects of SM still lack an explanation and the presence of **additional fundamental laws/particles (BSM)** is suggested by:

- experimental evidence for neutrino oscillations
- the matter/antimatter asymmetry in the Universe
- the necessary existence of the dark matter



The three frontiers



Technologic challenges at the Energy Frontier

- High energy/Large accelerators
- Detectors sensitive to signals
- Very strong background rejection
- Reconstruction of many complex events in reasonable amount of time
- Big amount of data to analyze
- Simulation

Large Hadron Collider

- World leading particle accelerator
  - 26.7 km long, 100m underground in Geneva countryside
- p-p or heavy ions collider
- Center-of-mass energy 13TeV
- Rate of physics process *i*

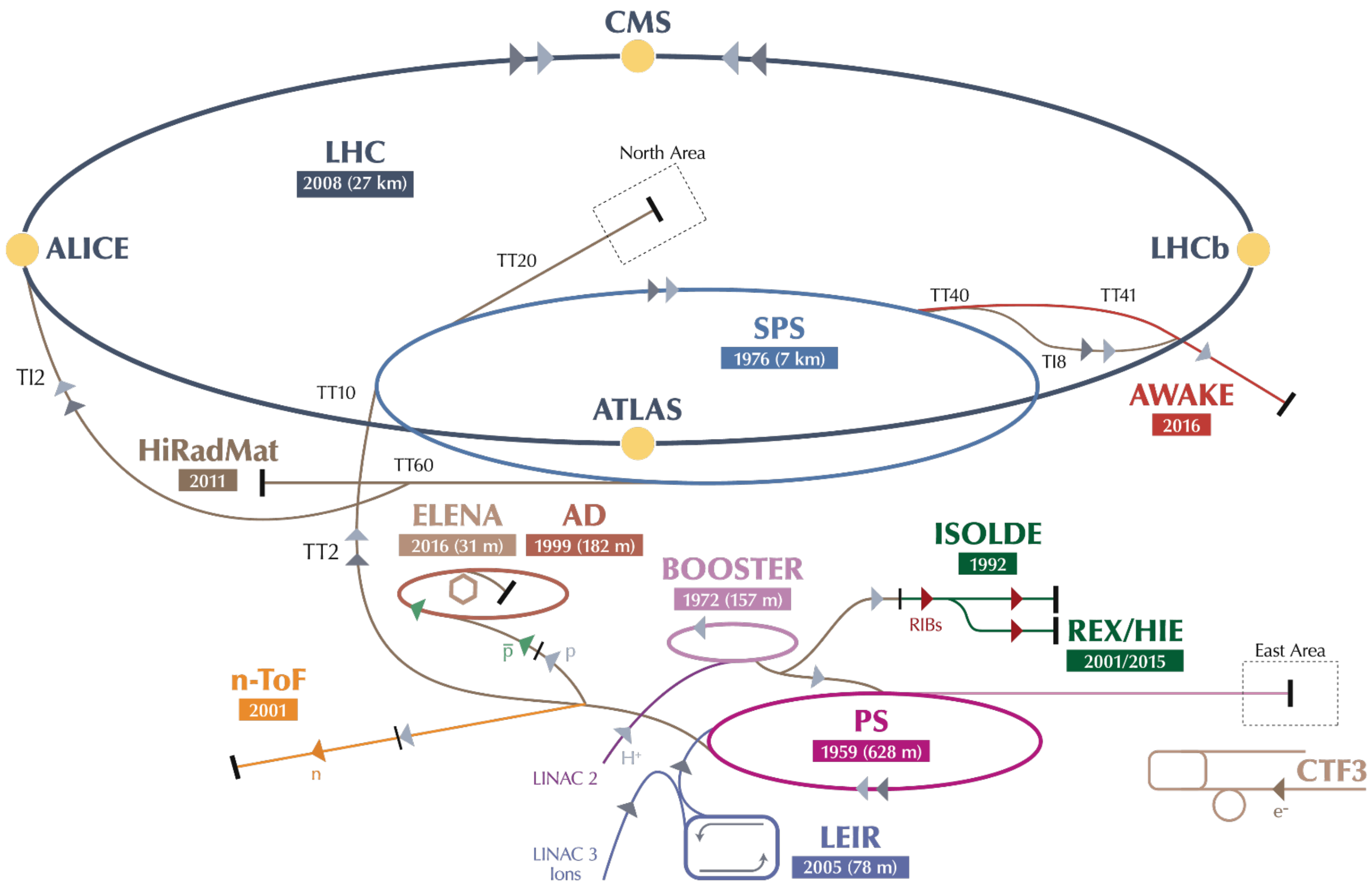
$$\frac{dN_i}{dt} = L\sigma_i$$

- Pile-up

$$\mu = \frac{L\sigma_{\text{inel}}}{f_{BX}}$$

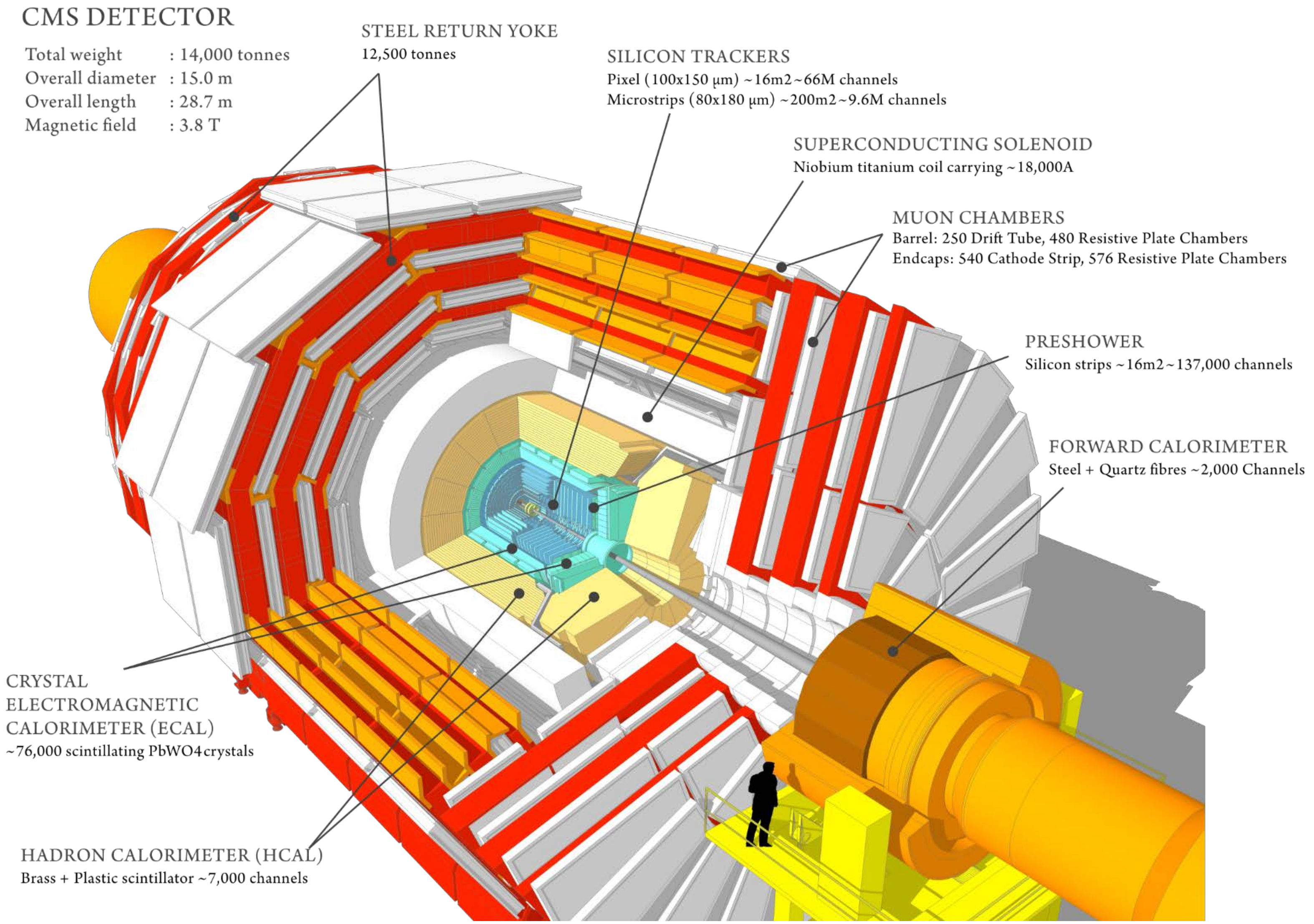
- Luminosity

$$L = F\frac{N_p^2 f k}{4\pi\sigma_x\sigma_y}$$



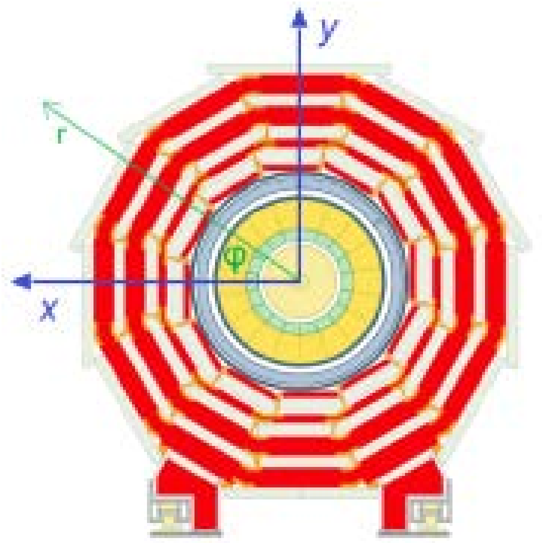
CMS Detector

- Cylindrical structure, hermetic around the beamline
- Produces ~1PB/s
  - To be reduced to O(100PB/y)
- High granularity and low occupancy
  - Sensitivity to signal
- Primary goals:
  - Precision measurements of the observed Higgs boson and its compatibility with the Standard Model
  - Provide evidence of physics beyond the Standard Model

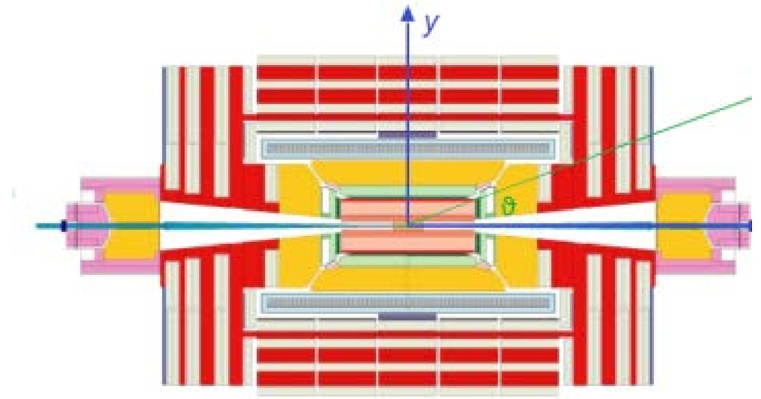

CMS DETECTOR
Total weight : 14,000 tonnes
Overall diameter : 15.0 m
Overall length : 28.7 m
Magnetic field : 3.8 T
STEEL RETURN YOKE
12,500 tonnes
SILICON TRACKERS
Pixel (100x150 μm) ~16m2 ~66M channels
Microstrips (80x180 μm) ~200m2 ~9.6M channels
SUPERCONDUCTING SOLENOID
Niobium titanium coil carrying ~18,000A
MUON CHAMBERS
Barrel: 250 Drift Tube, 480 Resistive Plate Chambers
Endcaps: 540 Cathode Strip, 576 Resistive Plate Chambers
PRESHOWER
Silicon strips ~16m2 ~137,000 channels
FORWARD CALORIMETER
Steel + Quartz fibres ~2,000 Channels
CRYSTAL ELECTROMAGNETIC CALORIMETER (ECAL)
~76,000 scintillating PbWO4 crystals
HADRON CALORIMETER (HCAL)
Brass + Plastic scintillator ~7,000 channels

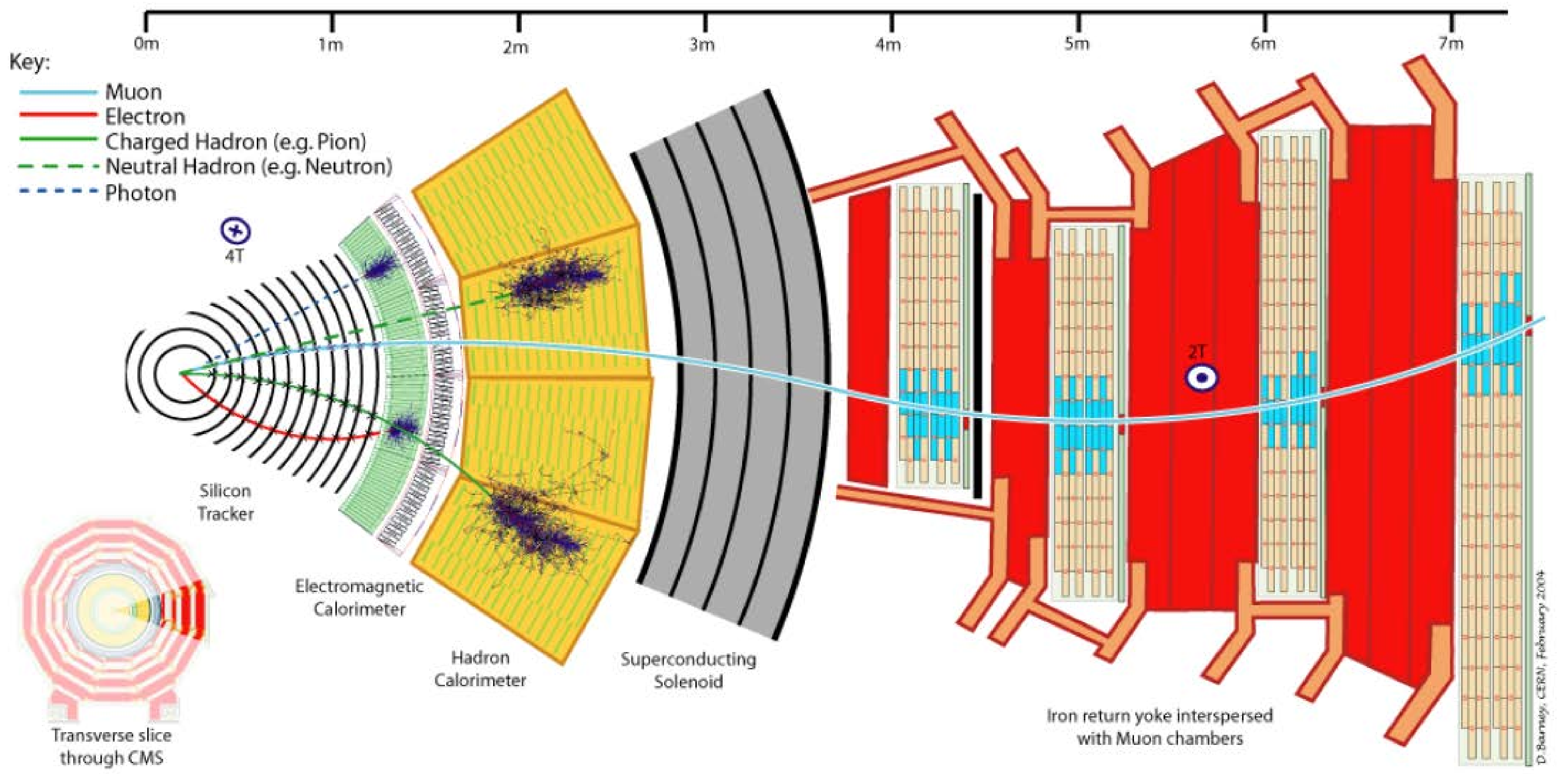
Structure of the CMS Detector

Transverse plane



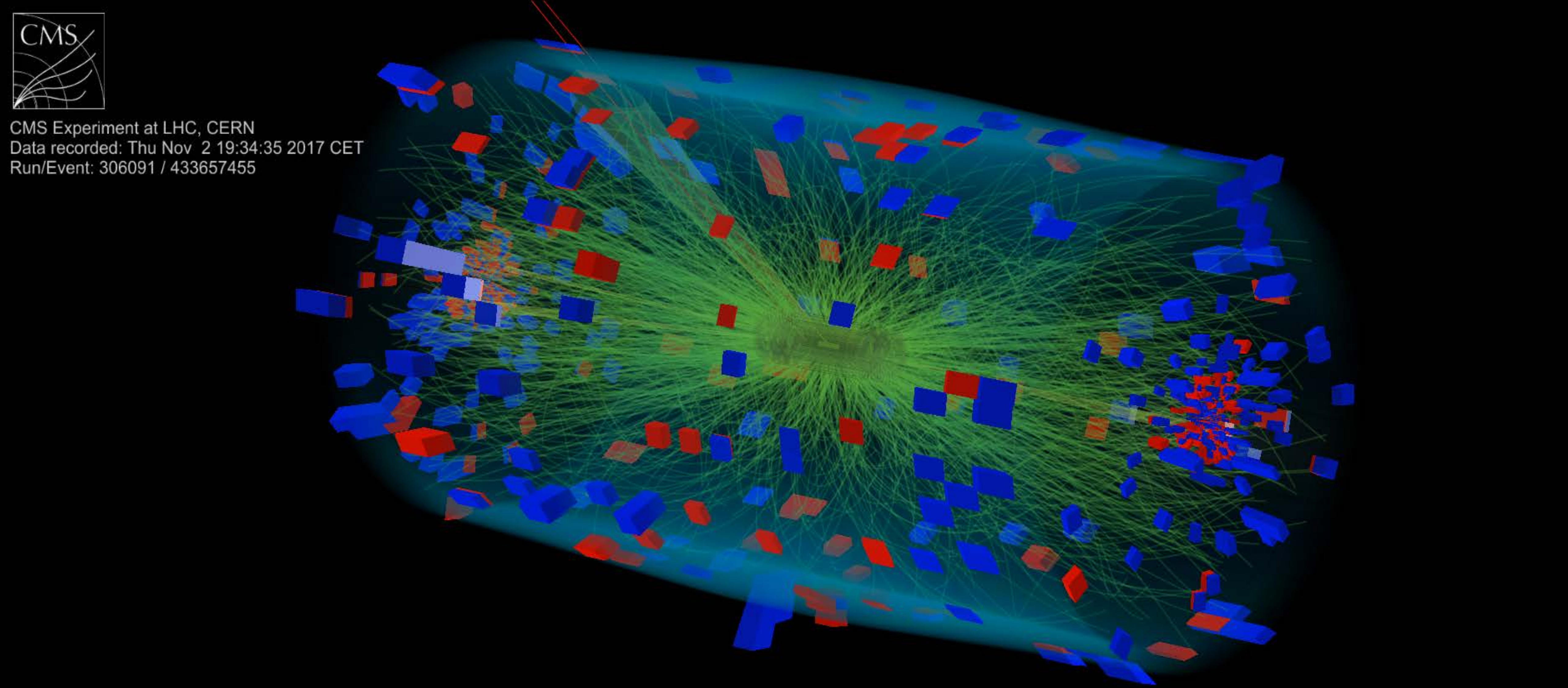
Longitudinal plane





0m
1m
2m
3m
4m
5m
6m
7m
Key:
Muon
Electron
Charged Hadron (e.g. Pion)
Neutral Hadron (e.g. Neutron)
Photon
4T
2T
Silicon Tracker
Electromagnetic Calorimeter
Hadron Calorimeter
Superconducting Solenoid
Iron return yoke interspersed with Muon chambers
Transverse slice through CMS
D.Barney, CERN, February 2004

Only a couple of weeks ago…


CMS
CMS Experiment at LHC, CERN
Data recorded: Thu Nov 2 19:34:35 2017 CET
Run/Event: 306091 / 433657455

Two-stages event selection strategy

Trigger System

- Reduce input rate (**40 MHz**) to a data rate (**~1 kHz**) that can be stored, reconstructed and analyzed Offline maximizing the physics reach of the experiment

Level 1 Trigger

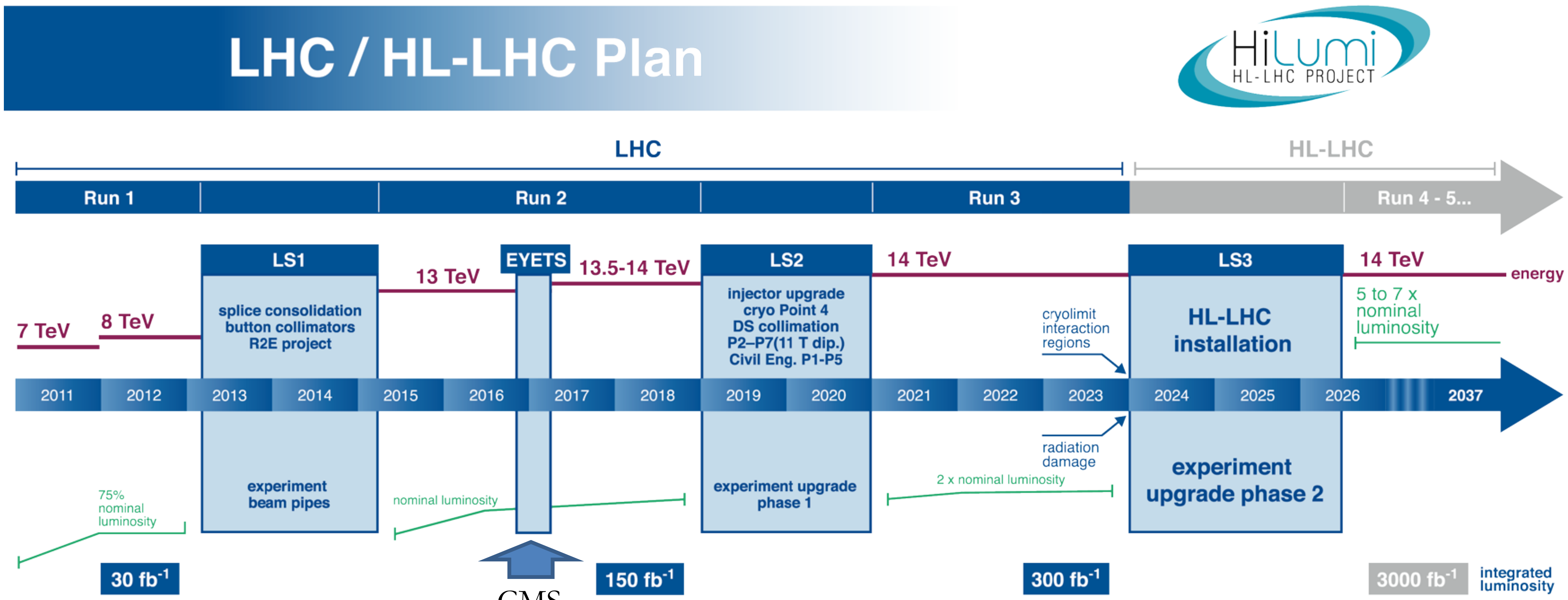
- coarse readout of the Calorimeters and Muon detectors
- implemented in custom electronics, ASICs and FPGAs
- output rate limited to **100 kHz** by the readout electronics

High Level Trigger

- readout of the whole detector with full granularity
- based on the CMS software, running on 22,000 CPU cores
- output rate limited to an average of **~1 kHz** by the Offline resources

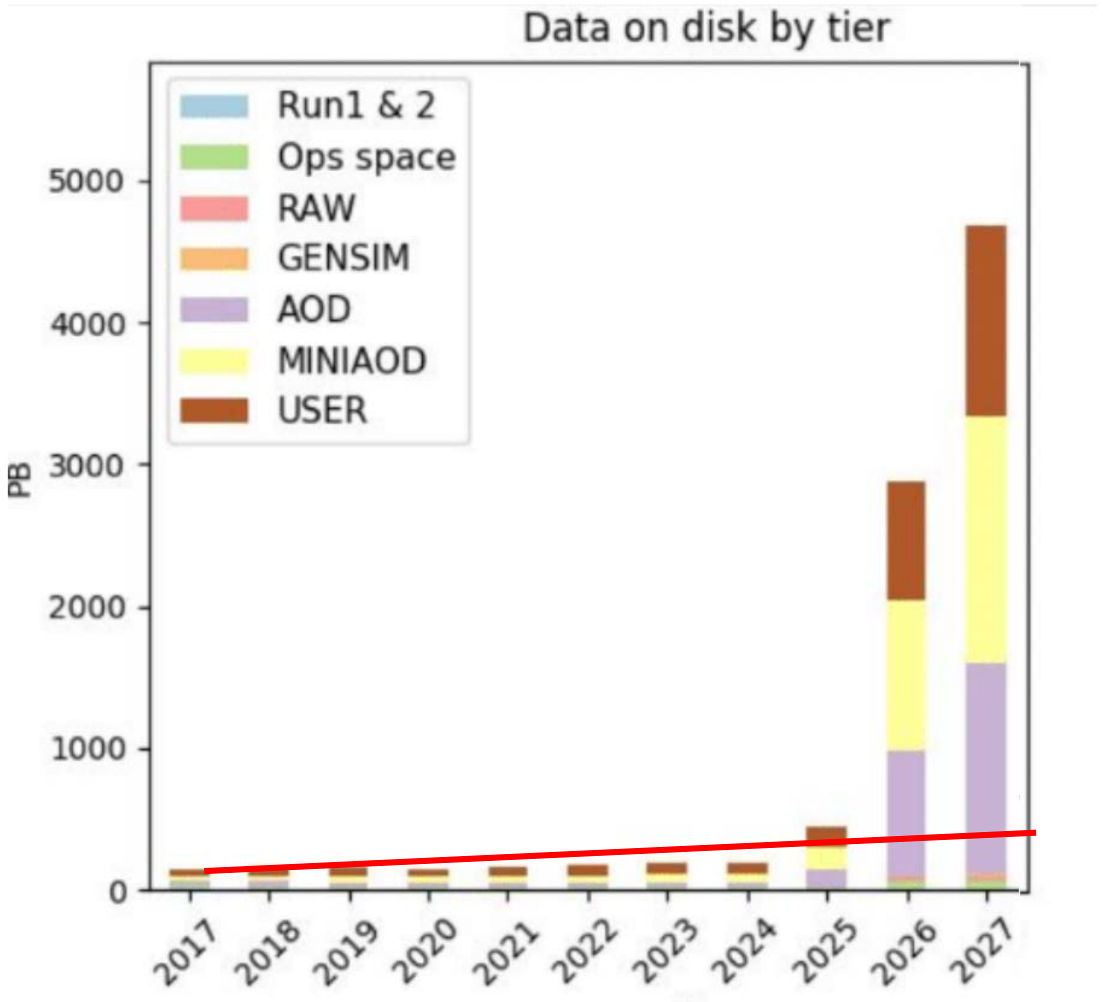
0.000 000 044s
Trigger Level 1
Ultra-fast event selection
Sélection ultra rapide des événements

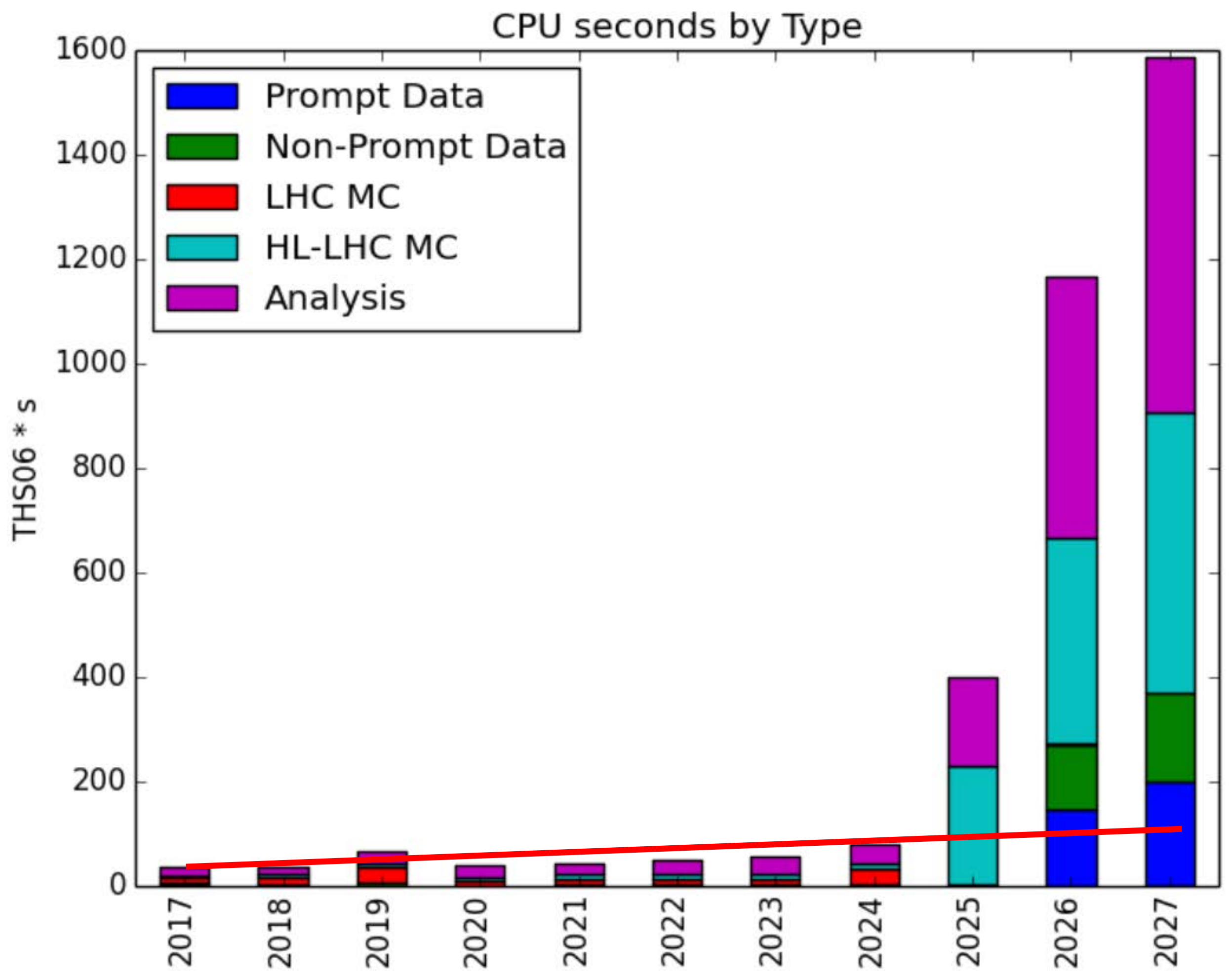
CMS and LHC Upgrade Schedule



CMS
pixel detector upgrade

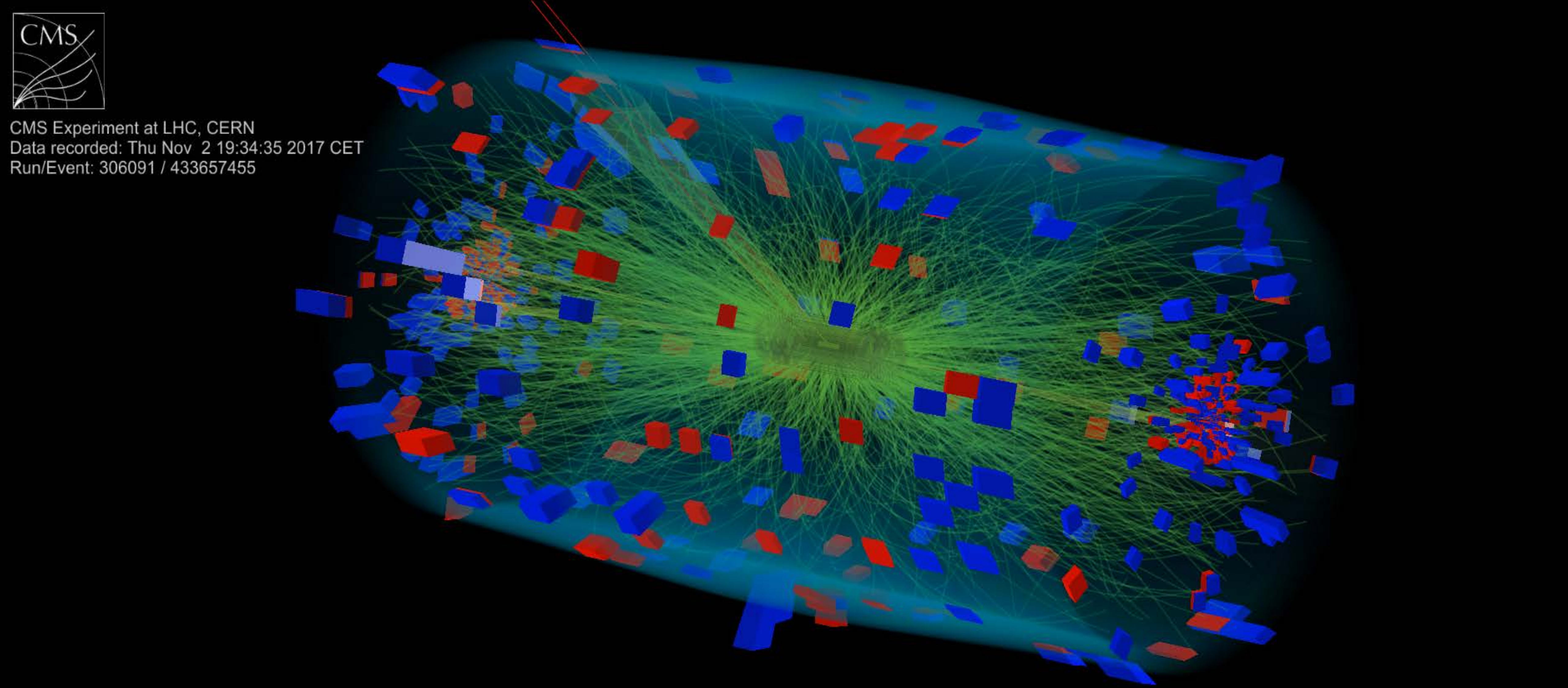
What we need to do…





…with flat budget

- This poses some constrain on:
  - Throughput: Events/s
  - Trigger efficiency
  - Events/CHF
  - Events/Joule


CMS Experiment at LHC, CERN
Data recorded: Thu Nov 2 19:34:35 2017 CET
Run/Event: 306091 / 433657455

Two-stages event selection strategy

Trigger System

- Reduce input rate (**40 MHz**) to a data rate (**~1 kHz**) that can be stored, reconstructed and analyzed Offline maximizing the physics reach of the experiment

Level 1 Trigger

- coarse readout of the Calorimeters and Muon detectors
- implemented in custom electronics, ASICs and FPGAs
- output rate limited to **100 kHz** by the readout electronics

High Level Trigger

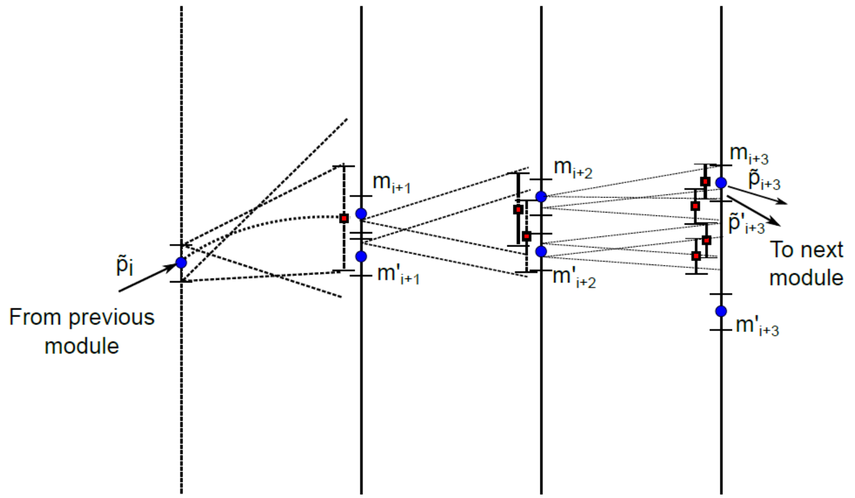
- readout of the whole detector with full granularity
- based on the CMS software, running on 22,000 CPU cores
- output rate limited to an average of **~1 kHz** by the Offline resources

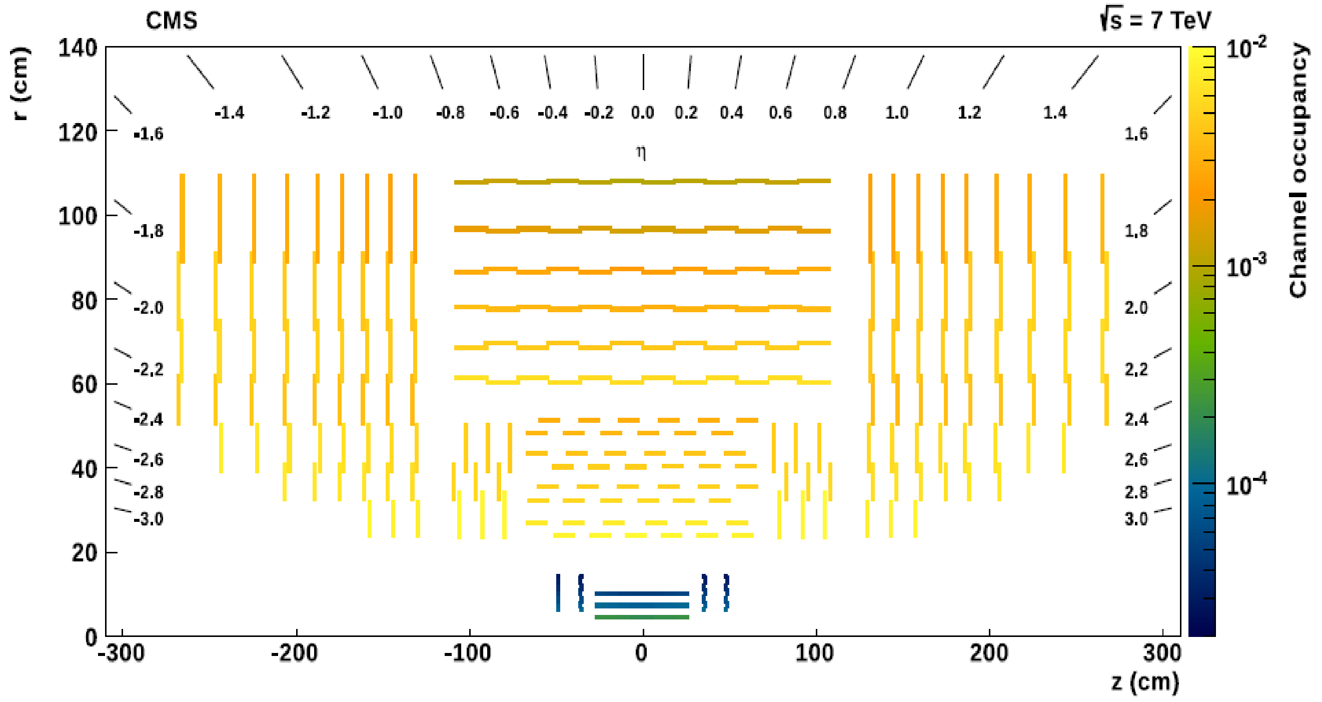
High-Level Trigger
0.000 000 044s
Trigger Level 1
Ultra-fast event selection
Sélection ultra rapide des événements

Track Reconstruction

Combinatorial Kalman Filter

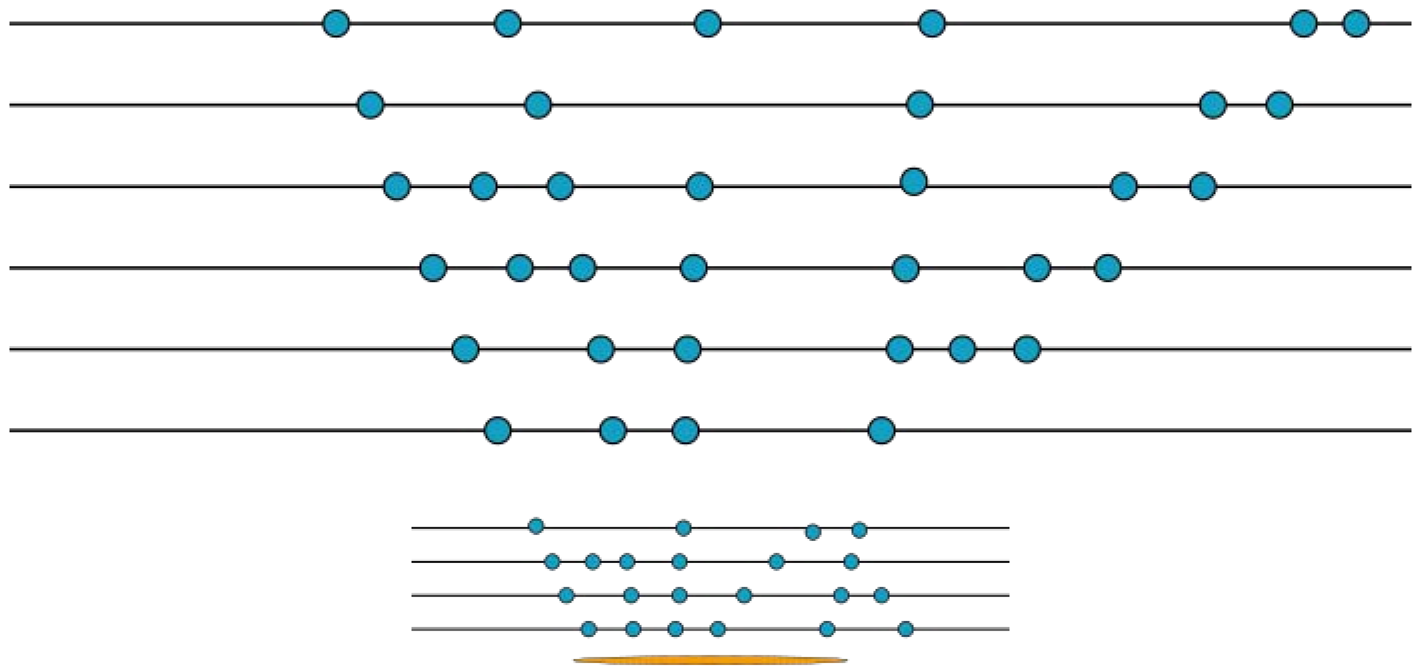
- Track parameterization:
  - $(1/p, \theta, \phi, d_{xy}, d_{sz})$
- When multiple measurements are compatible with the propagated state vector tracking becomes a combinatorial problem
- Track seed important
  - **Reduces the initial number of combinations to update and propagate**
  - Provides an initial estimate of the track parameters to be used as initial state vector in the track building
  - Pixel Detector for low occupancy
  - Seed is computed in the pixel wrt to a seeding region (origin coordinates and size, and minimum transverse momentum allowed)





Iterative tracking

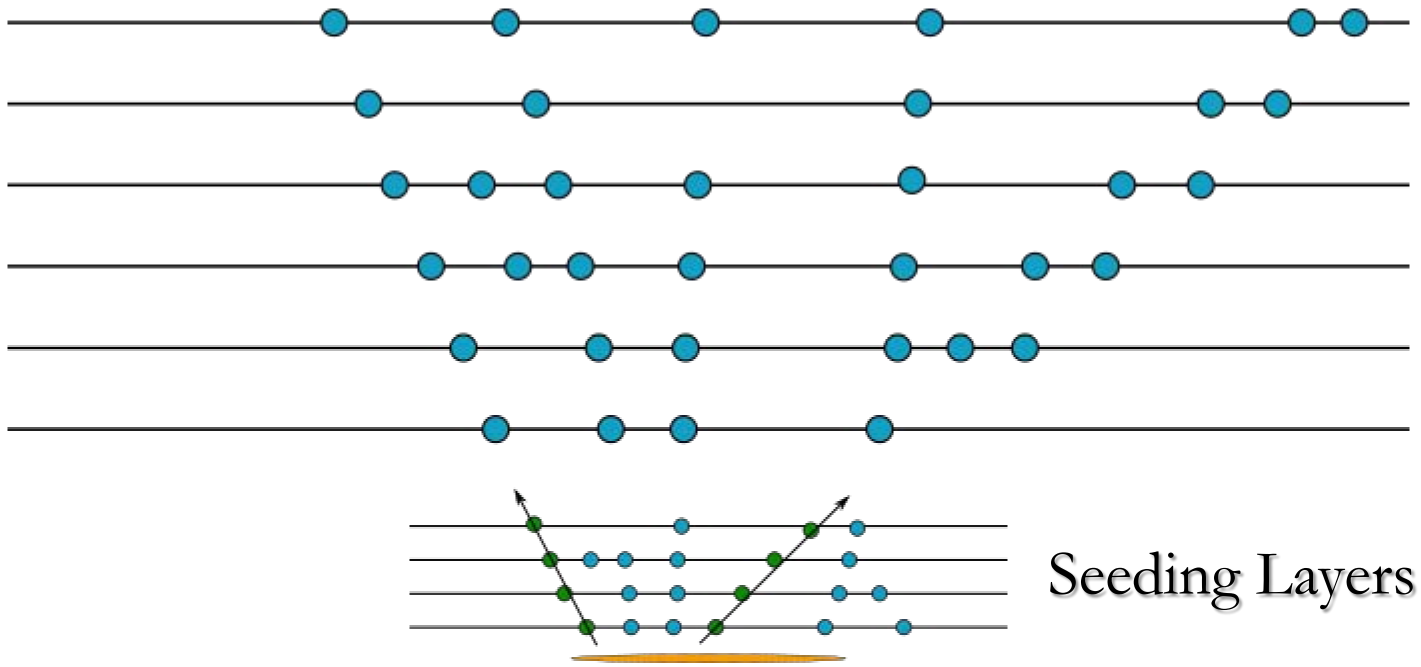
- In order to reduce combinatorial complexity, track reconstruction is iterative
- Constrains on seeding region and seeding layers become looser at each iteration



Hits are formed

Iterative tracking

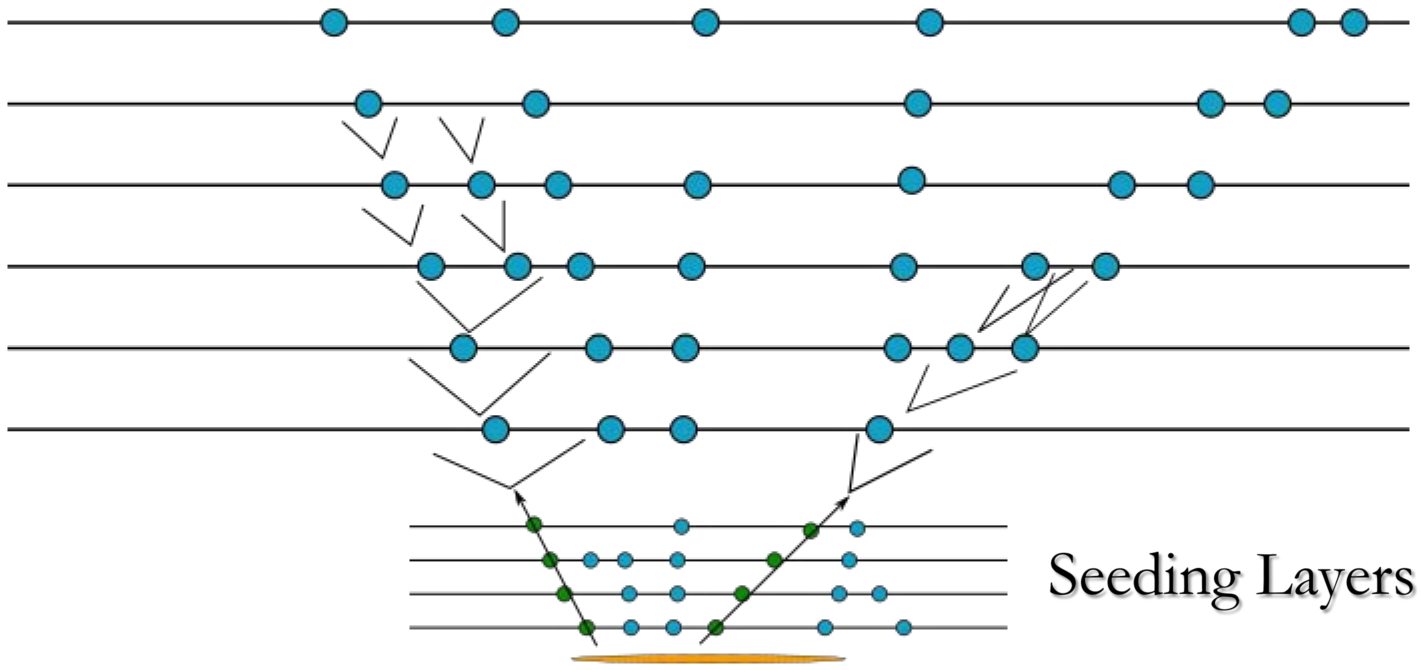
- In order to reduce combinatorial complexity, track reconstruction is iterative
- Constrains on seeding region and seeding layers become looser at each iteration



A seeding region is defined and seeds are created

Iterative tracking

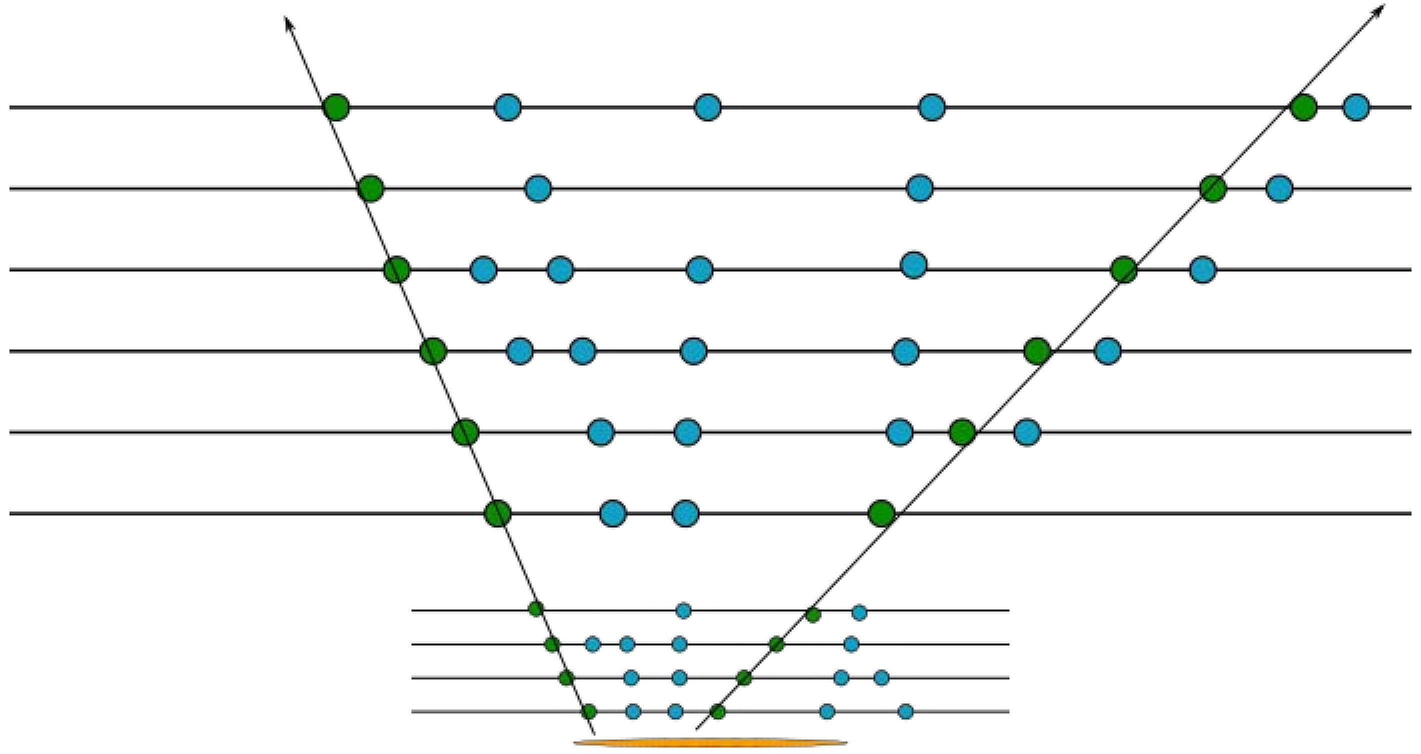
- In order to reduce combinatorial complexity, track reconstruction is iterative
- Constrains on seeding region and seeding layers become looser at each iteration



Tracks are built starting from seeds

Iterative tracking

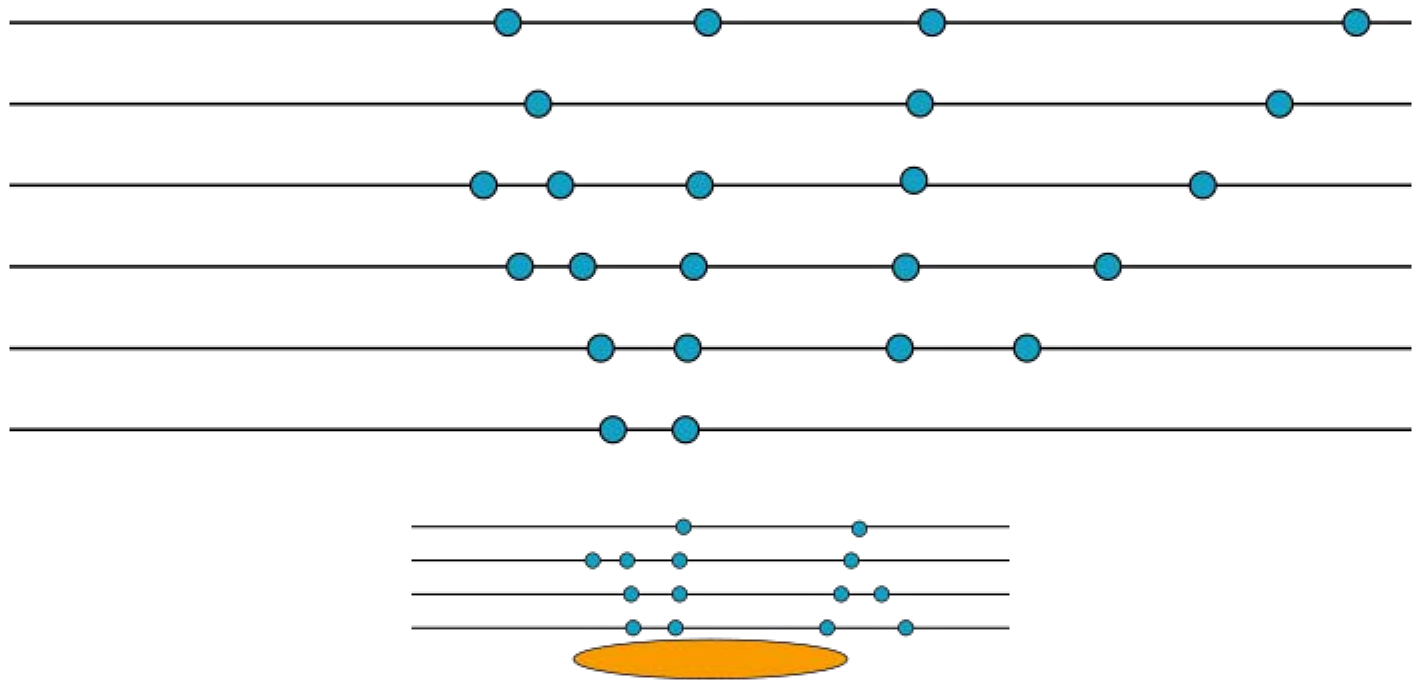
- In order to reduce combinatorial complexity, track reconstruction is iterative
- Constrains on seeding region and seeding layers become looser at each iteration



Track fit and selection

Iterative tracking

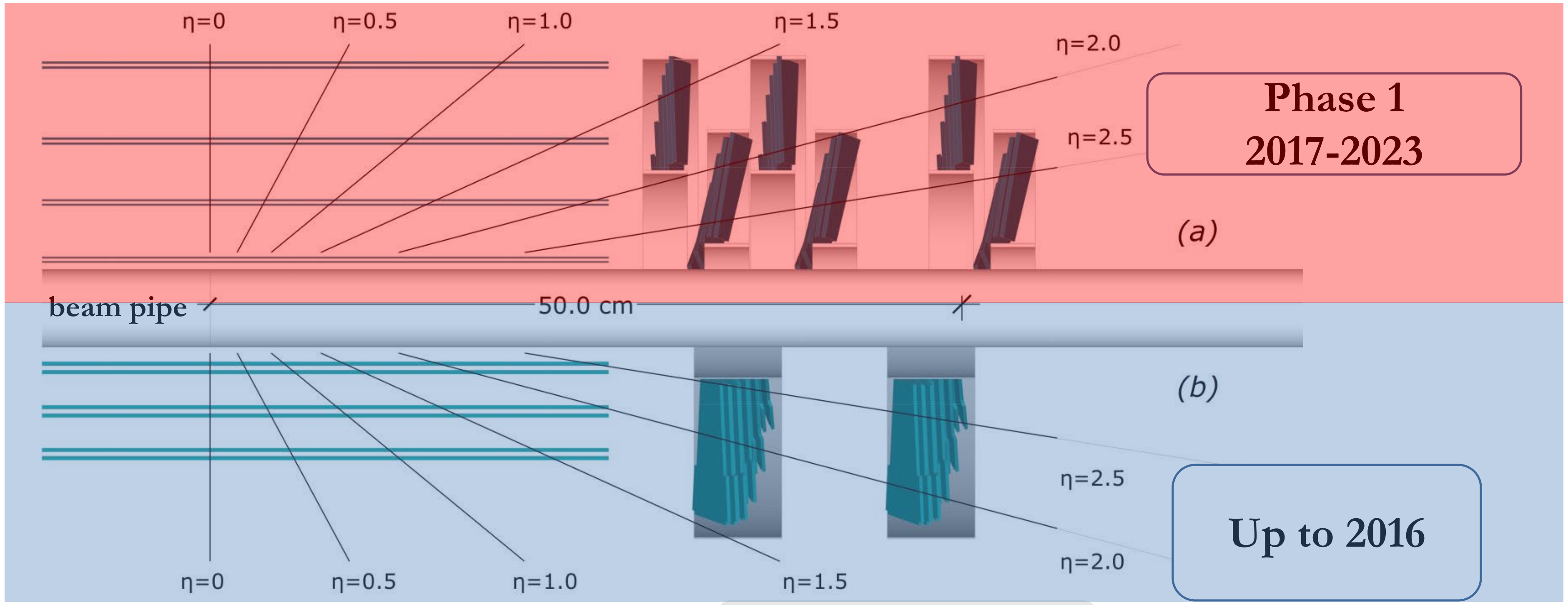
- In order to reduce combinatorial complexity, track reconstruction is iterative
- Constrains on seeding region and seeding layers become looser at each iteration



Hits belonging to selected tracks are masked.
A new seeding configuration is used and a new iteration starts

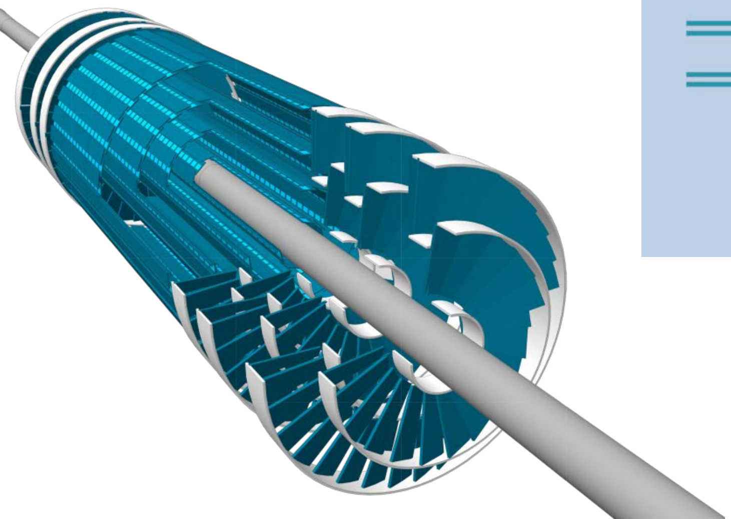
Phase 1 Pixel detector

Starting from 2017 the already complex online and offline track reconstruction has to deal not only with a much more crowded environment but also with data coming from a more complex detector.



Barrel Pixel (BPIX)

Forward Pixel (FPIX)



$$\eta = -\ln \tan(\theta/2)$$

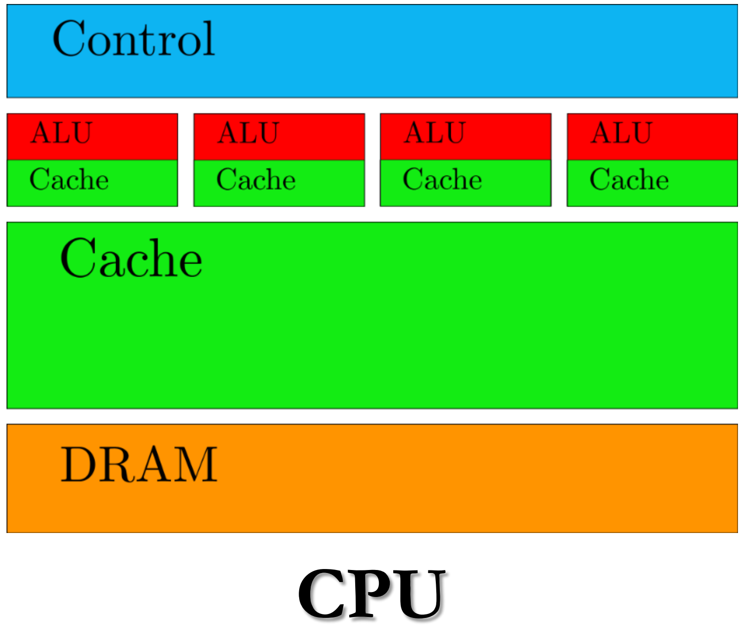
Pixel Tracks

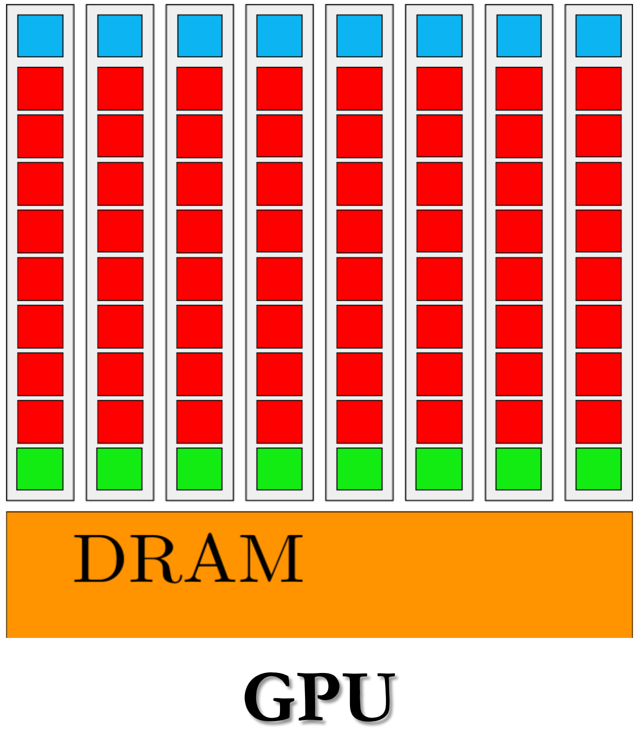
- Evaluation of Pixel Tracks combinatorial complexity is dominated by pileup and is one of the main bottlenecks of the High-Level Trigger and offline reconstruction execution times.
- The CMS HLT farm and its offline computing infrastructure cannot rely anymore on an exponential growth of frequency guaranteed by the manufacturers
- Hardware and algorithmic solutions have been studied in the context of this thesis work

A Parallel Hit-Chain Maker based on Cellular Automata

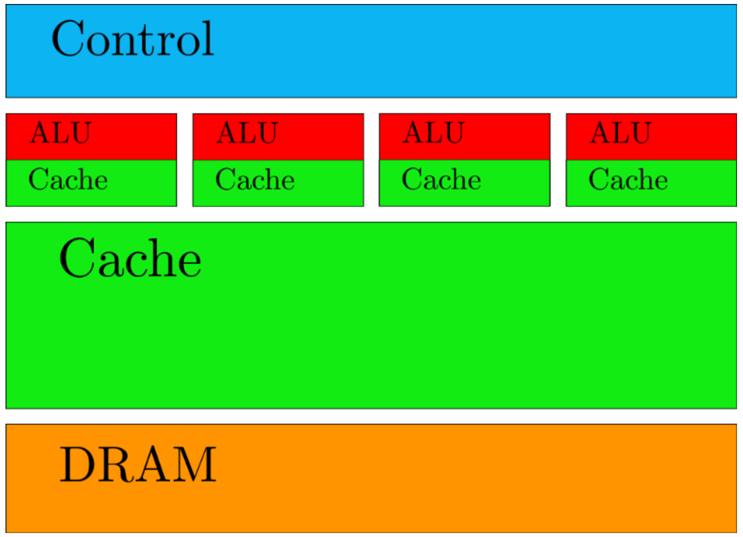
Game of Tracks

CPU vs GPU architectures





CPU vs GPU architectures



CPU

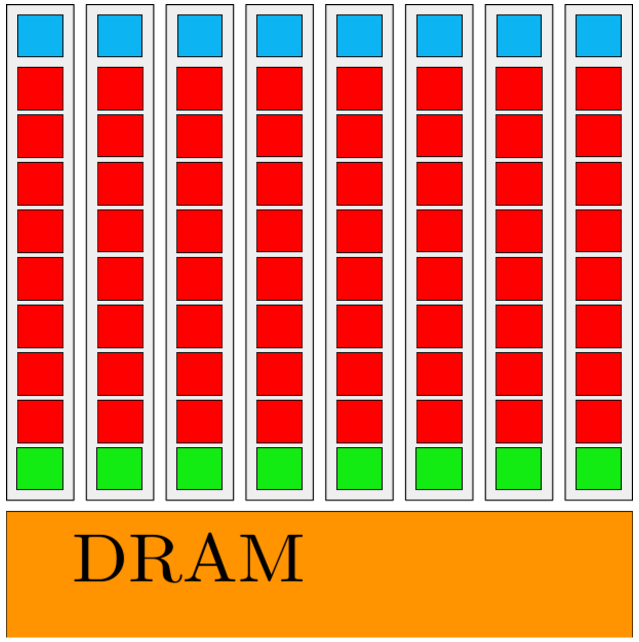
- Large caches (slow memory accesses to quick cache accesses)
- Powerful ALUs
- Low bandwidth to memory (tens GB/s)

In CMS:

- One event per core, thanks to independency of events
- Memory footprint a issue

CPU vs GPU architectures

- Many Streaming Multiprocessors execute kernels (aka functions) using hundreds of threads concurrently
- High bandwidth to memory (up to 1TB/s)
- Number of threads in-fly increases with each generation
- In CMS:
  - unroll and offload each event's combinatorics to many threads in parallel



GPU

Compute intensity

P100

- 7.8 TFLOPS DPFP peak throughput
- 900 GB/s peak off-chip HBM2 memory access bandwidth
  - 112.5 billion DPFP operands per second

Intel KNL

- 3 TFLOPS DPFP peak throughput
- 500 GB/s High BW memory + DDR4

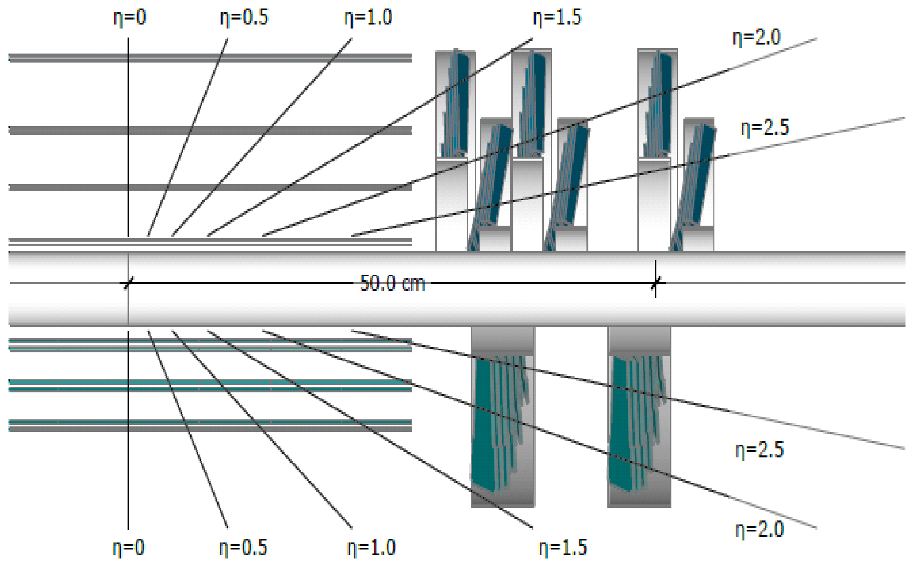
- To achieve peak throughput, a program must perform 7,800/112.5 = ~70 FP arithmetic operations for each double precision operand value fetched from off-chip memory

Bandwidth

Bandwidth

From RAW to Tracks during run 3

- Profit from the end-of-year upgrade of the Pixel to redesign the seeding code from scratch
  - Exploiting the information coming from the 4th layer would improve efficiency, b-tag, IP resolution
- Trigger avg latency should stay within 220ms
- Reproducibility of the results (bit-by-bit equivalence CPU-GPU)
- Integration in the CMS software framework



- Ingredients:
  - Massive parallelism within the event
  - Independence from thread ordering in algorithms
  - Avoid useless data transfers and transformations
  - Simple data formats optimized for parallel memory access
- Result:
  - A GPU based application that takes RAW data and gives Tracks as result

Algorithm Stack

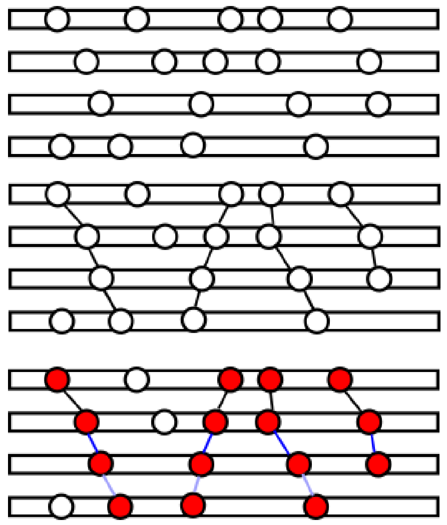
Input, size linear with PU

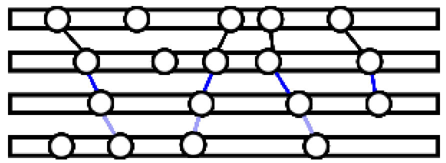
Raw to Digi

Hits - Pixel Clusterizer

Hit Pairs

CA-based Hit Chain Maker

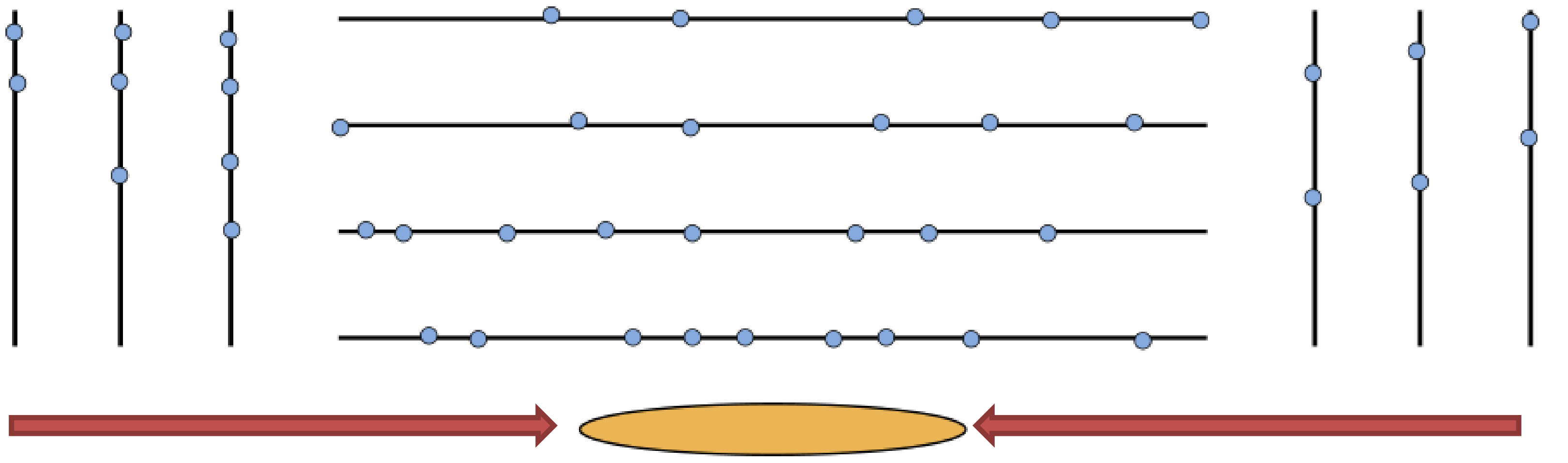




Output, size ~linear with PU + dependence on fake rate

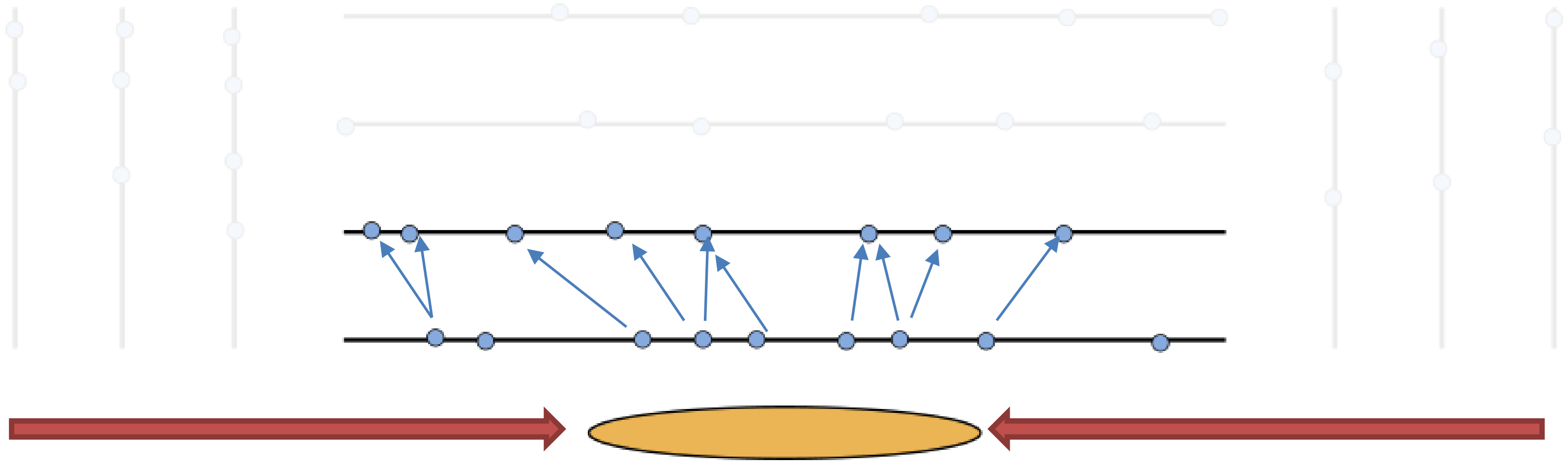
Existing Triplet Propagation Algorithm

- Hits on different layers
- Need to match them and create quadruplets
- Create a modular pattern and reapply it iteratively



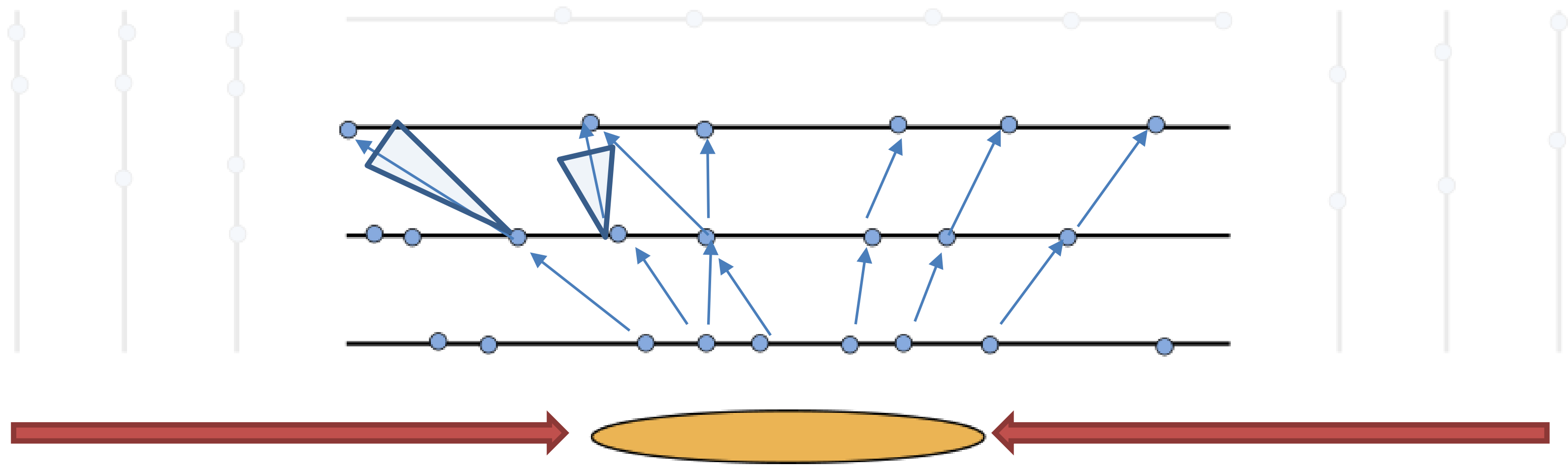
Existing Triplet Propagation Algorithm

- First create doublets from hits of pairs



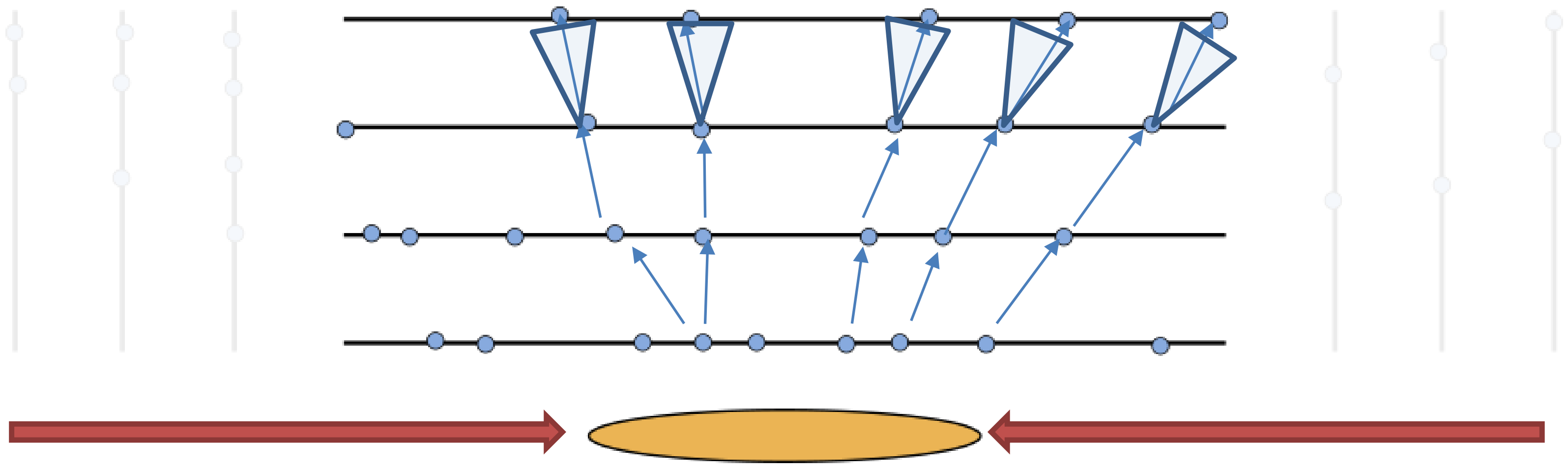
Existing Triplet Propagation Algorithm

- First create doublets from hits of pairs
- Take a third layer and propagate only the generated doublets



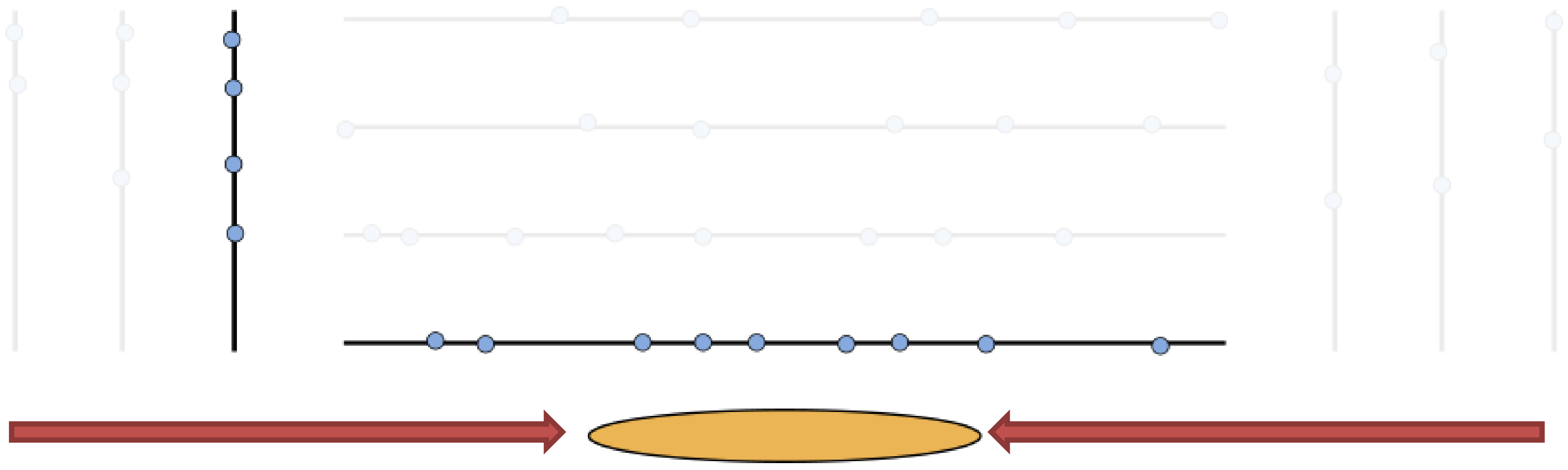
Existing Triplet Propagation Algorithm

- First create doublets from hits of pairs
- Take a third layer and propagate only the generated doublets
- Consider a fourth layer and propagate triplets
- Store found quadruplets and start from another pair of layers



Existing Triplet Propagation Algorithm

- First create doublets from hits of pairs
- Take a third layer and propagate only the generated doublets
- Consider a fourth layer and propagate triplets
- Store found quadruplets and start from another pair of layers



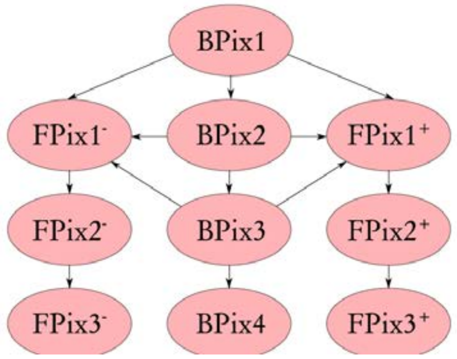
Existing Triplet Propagation Algorithm

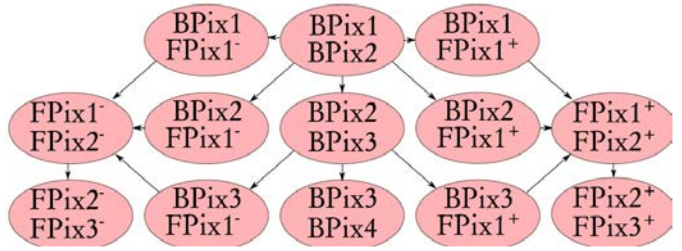
This kind of algorithm is not very suitable for GPUs:

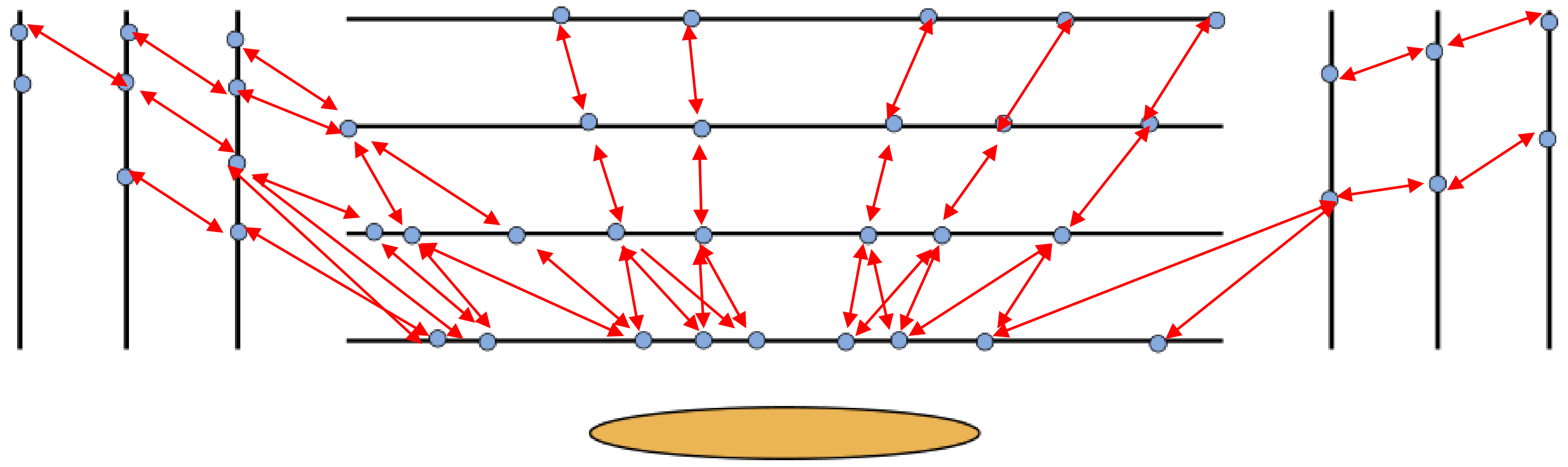
- Absence of massive parallelism
- Poor data locality
- Synchronizations due to iterative process
- Very Sparse and dynamic problem (that's the hardest part, still unsolved)
- Parallelization does not mean making a sequential algorithm run in parallel
  - It requires a deep understanding of the problem, renovation at algorithmic level, understanding of the computation and dependencies

Cellular Automaton-based Hit Chain-Maker

- The CA is a track seeding algorithm designed for parallel architectures
- It requires a list of layers and their pairings
  - A graph of all the possible connections between layers is created
  - Doublets aka Cells are created for each pair of layers (compatible with a region hypothesis)
  - Fast computation of the compatibility between two connected cells
  - No knowledge of the world outside adjacent neighboring cells required, making it easy to parallelize

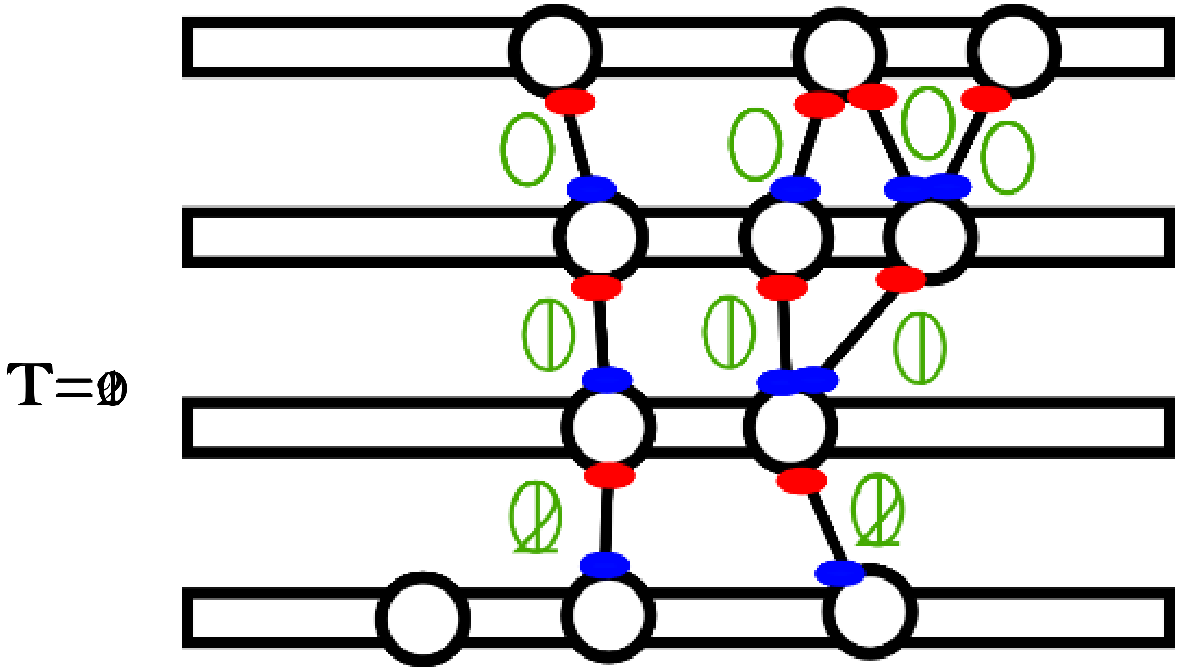






Evolution

- If two cells satisfy all the compatibility requirements they are said to be neighbors and their state is set to 0
- In the evolution stage, their state increases in discrete generations if there is an outer neighbor with the same state
- At the end of the evolution stage the state of the cells will contain the information about the length
- If one is interested in quadruplets, there will be surely one starting from a state 2 cell, pentuplets state 3, etc.



Tests and Results: HLT

Quality criteria

Efficiency:

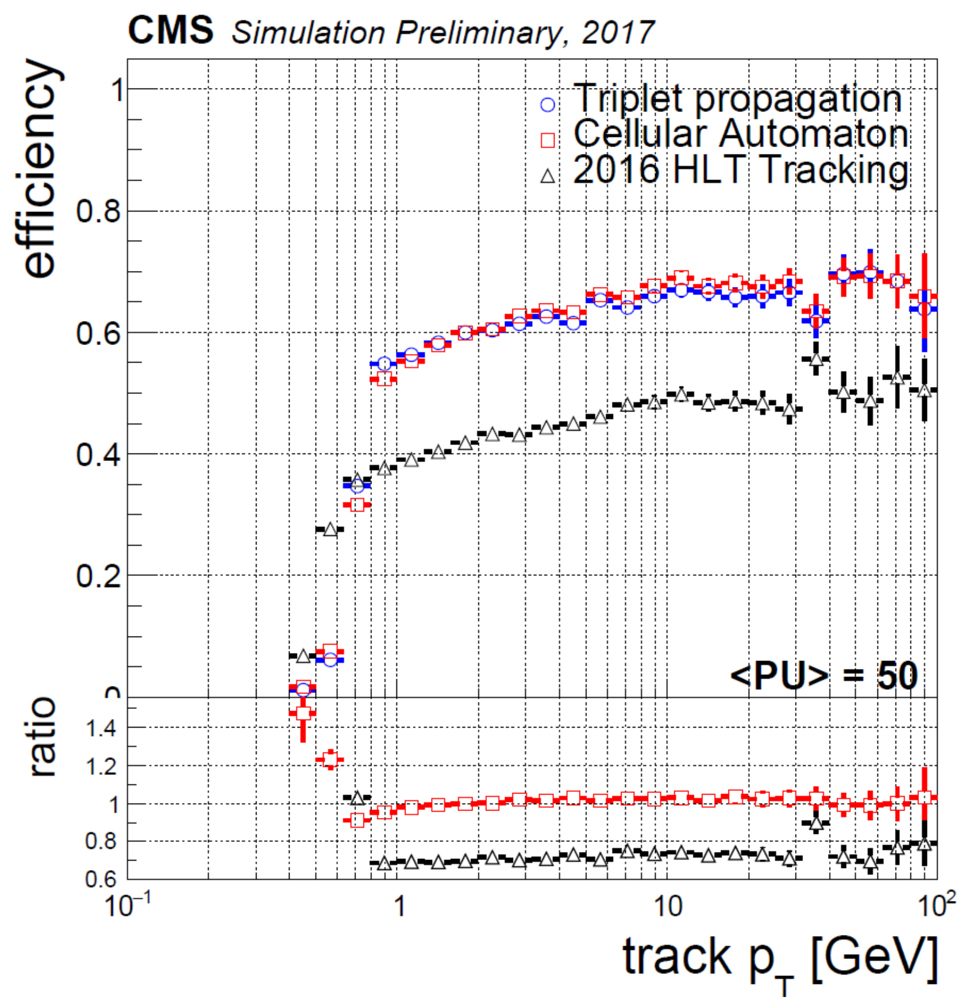
- indicates the fraction of the simulated tracks, N_{sim}, that have been associated with at least one reconstructed track, N_{rec}
  - Association with a simulated track if more than 75% of the hits that it contains come from the same simulated track

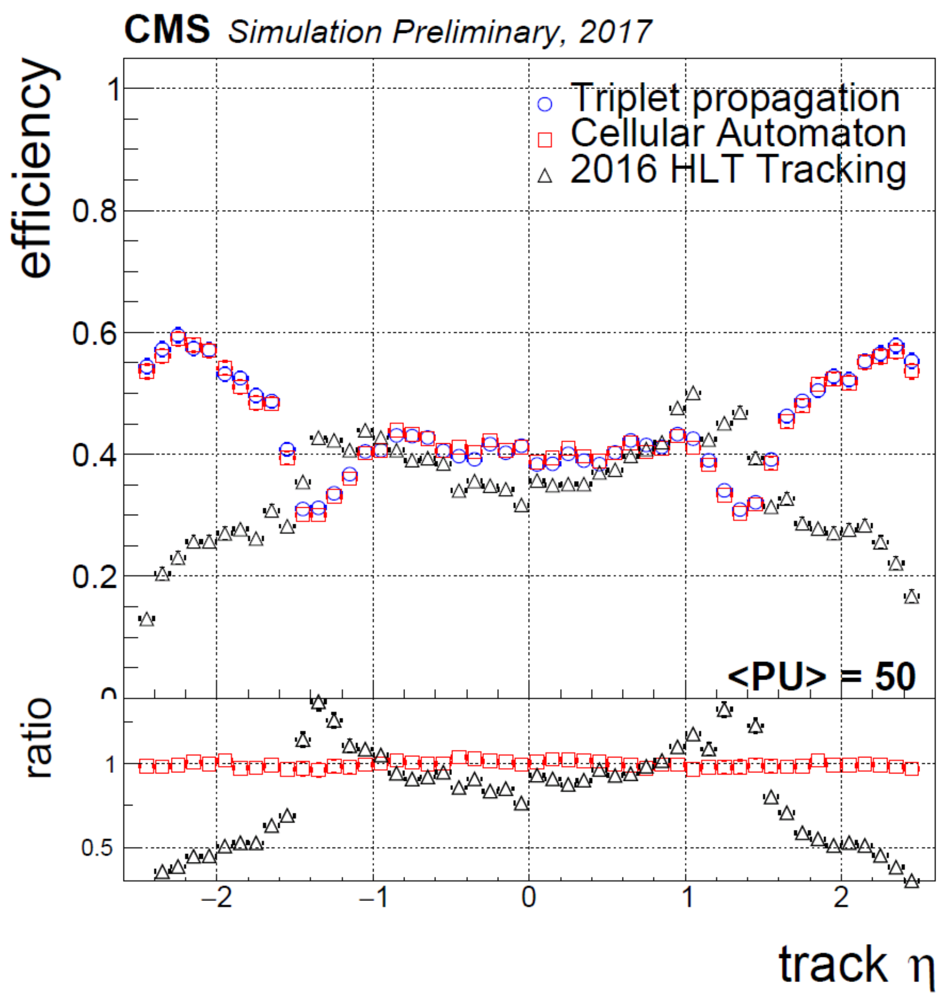
Fake rate:

- the fraction of all the reconstructed tracks which are not associated uniquely to a simulated track

Execution time

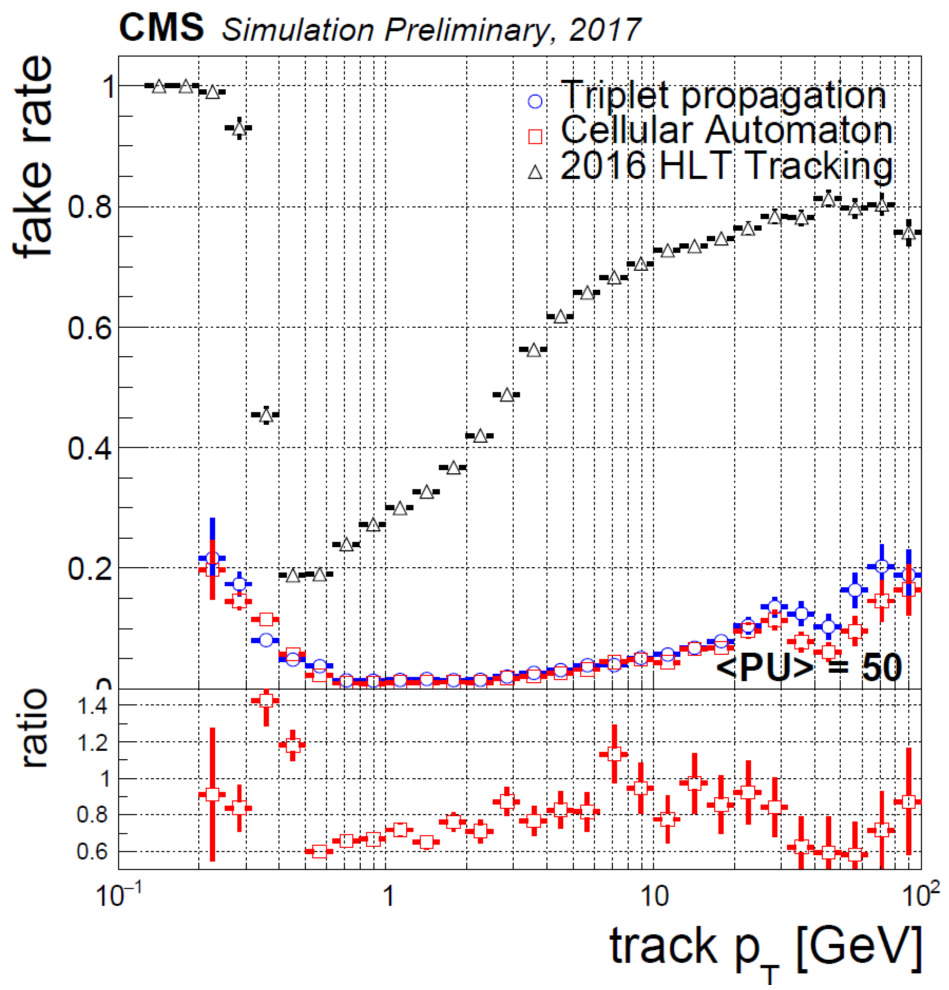
HLT: Simulated Physics Performance PixelTracks

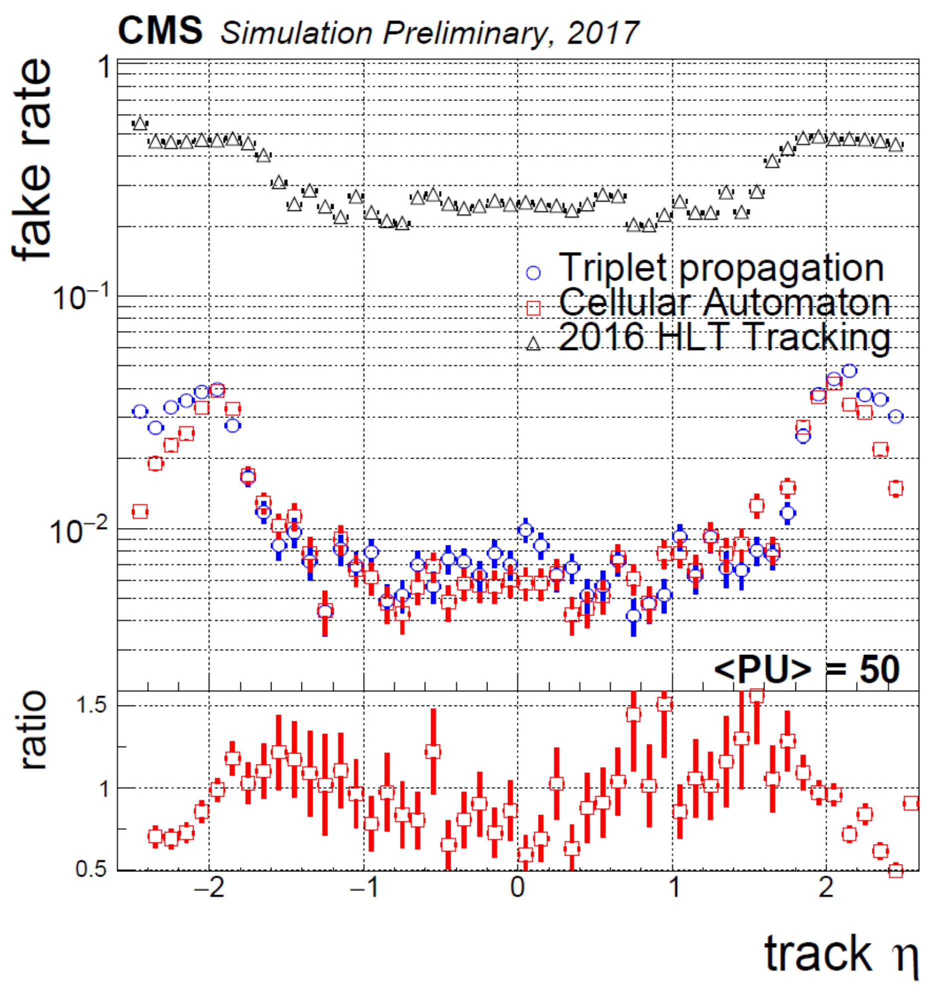




- CA Hit-Chain Maker tuned to have same efficiency as Triplet Propagation
- Efficiency significantly larger than 2016, especially in the forward region (|η|>1.5).

HLT: Simulated Physics Performance PixelTracks

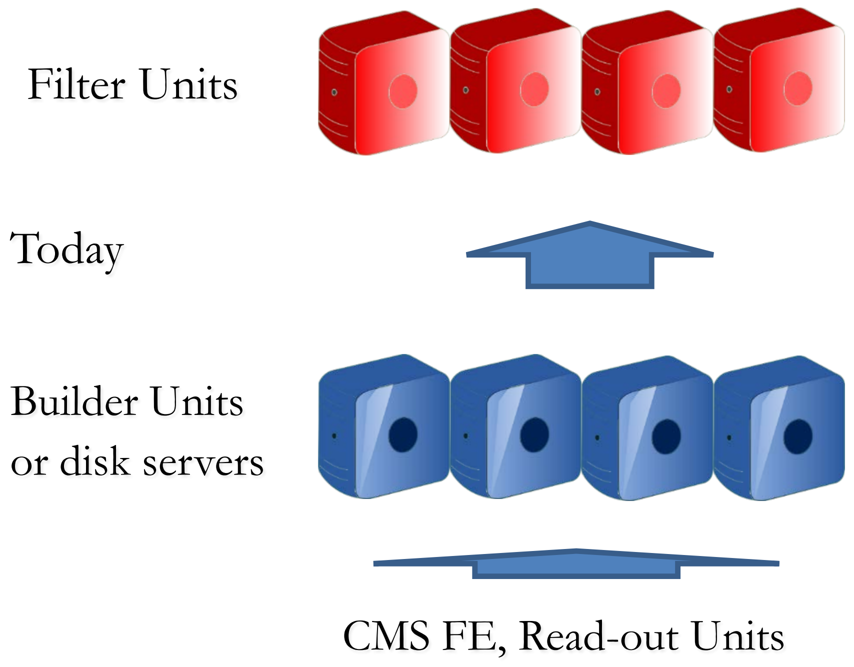




- Fake rate up to 40% lower than Triplet Propagation
- Two orders of magnitudes lower than 2016 tracking thanks to higher purity of quadruplets wrt to triplets

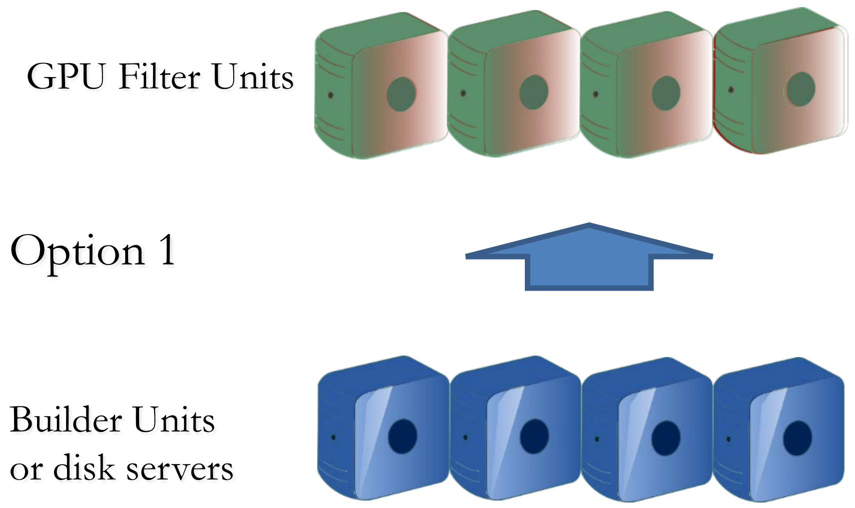
Integration in the Cloud and/or HLT Farm

- Different possible ideas depending on :
  - the fraction of the events running tracking
  - other parts of the reconstruction requiring a GPU



Integration in the Cloud/Farm

- Every FU is equipped with GPUs



- Rigid design

  + easiest to implement

  + offline reconstruction would benefit from this design

  - Requires common acquisition, dimensioning
- Smarter design
  - Requires concept of locality
  - Prefetching data
  - Run where data are
  - Move data where compute power is

From the framework side

- From the framework point of view:

BProducer

```
CopyToGPU(A,es)
Raw2Digi(A,B,es)
CopyFromGPU(B)
```

CProducer

```
CopyToGPU(B,es)
Clusterizer(B,C,es)
CopyFromGPU(C)
```

DProducer

```
CopyToGPU(C,es)
CPE(C,D,es)
CopyFromGPU(D)
```

Workflow A

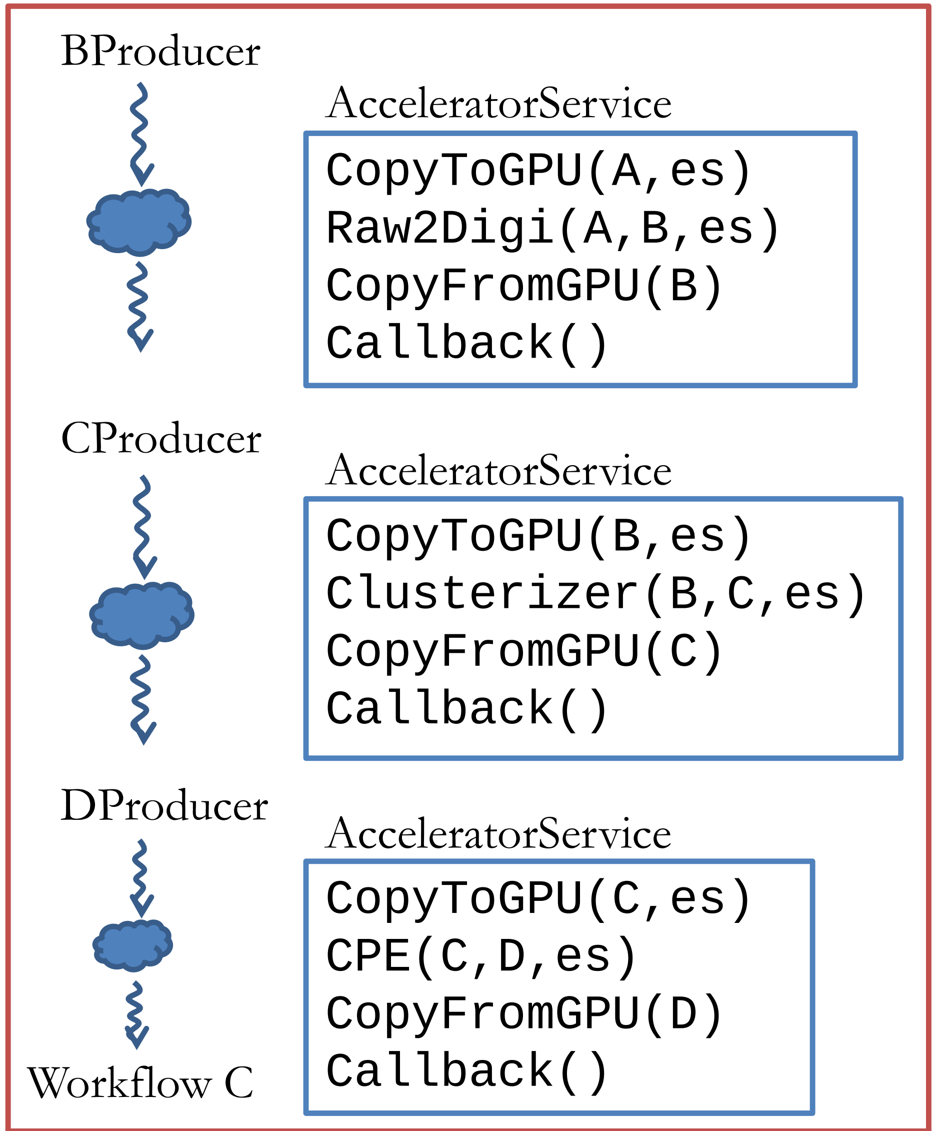
GPU PixelTracksProducer

```
CopyToGPU(A,es)
Raw2Digi(A,B,es)
Clusterizer(B,C,es)
CPE(C,D,es)
Doublets(D,E,es)
CA(E,F,es)
Fit(F,G,es)
CopyFromGPU(G)
```

Workflow B

Enhanced demonstrator

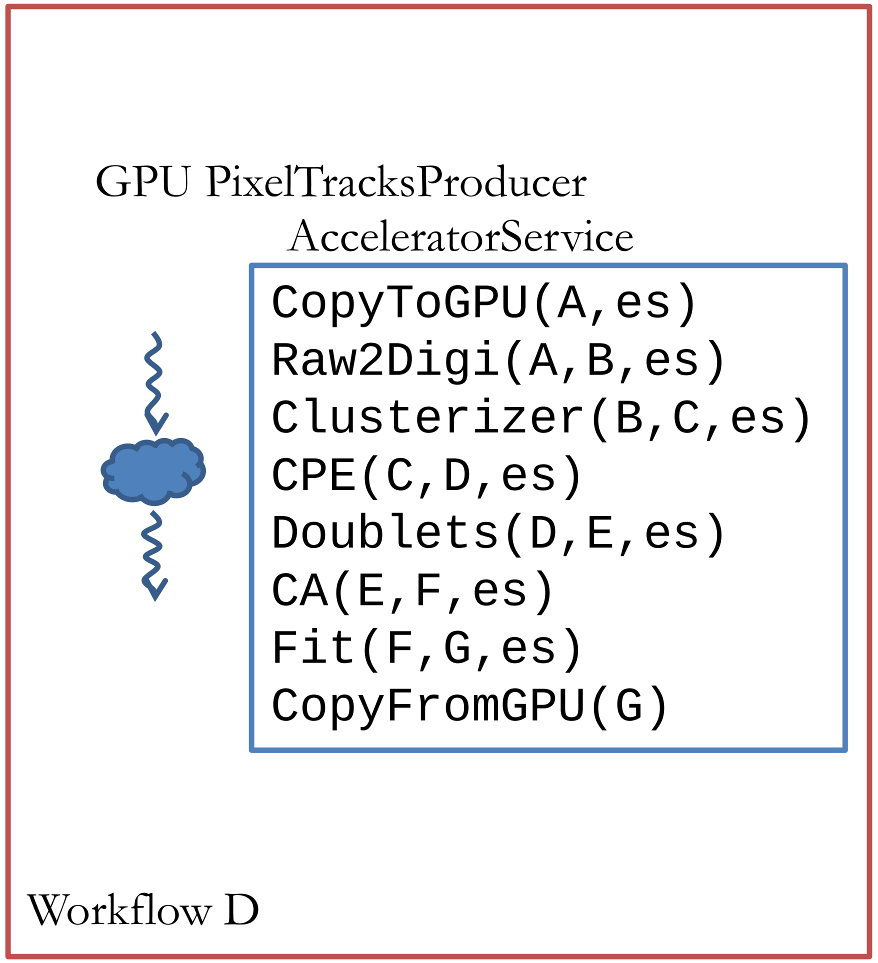
- Using external worker module



```
CopyToGPU(A,es)
Raw2Digi(A,B,es)
CopyFromGPU(B)
Callback()
```

```
CopyToGPU(B,es)
Clusterizer(B,C,es)
CopyFromGPU(C)
Callback()
```

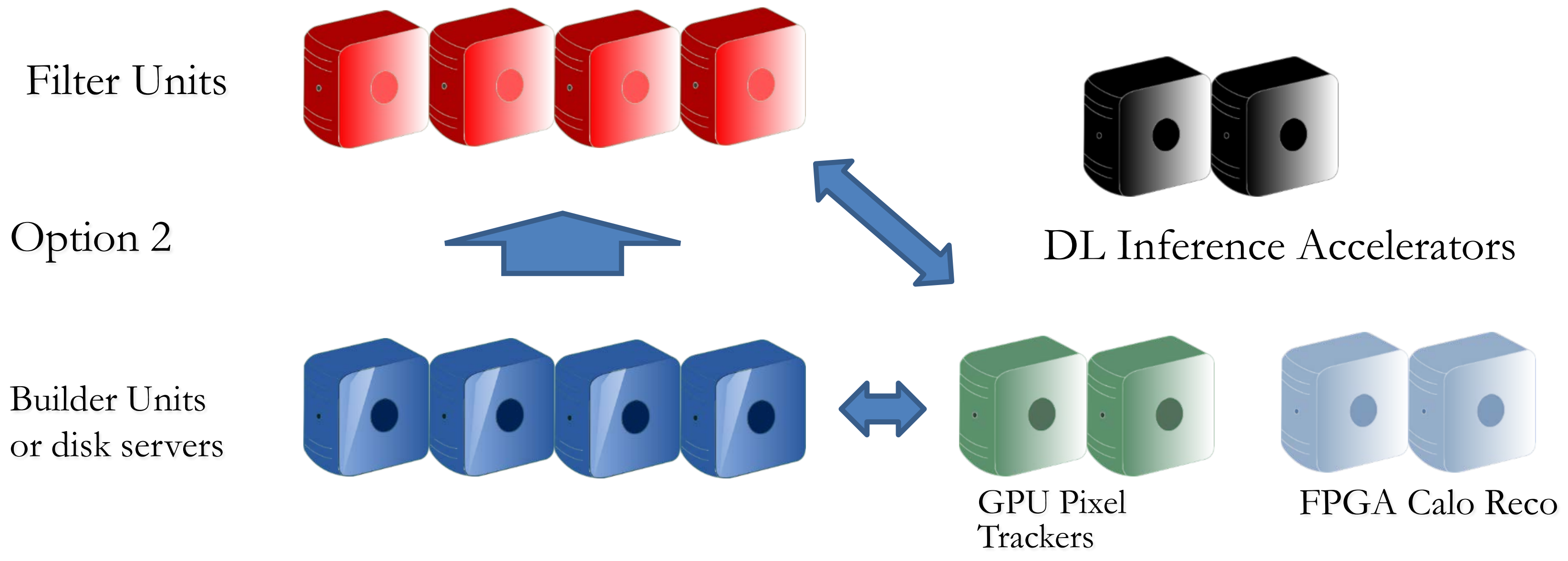
```
CopyToGPU(C,es)
CPE(C,D,es)
CopyFromGPU(D)
Callback()
```



```
CopyToGPU(A,es)
Raw2Digi(A,B,es)
Clusterizer(B,C,es)
CPE(C,D,es)
Doublets(D,E,es)
CA(E,F,es)
Fit(F,G,es)
CopyFromGPU(G)
```

Integration in the Cloud/Farm

- A part of the farm is dedicated to a high density GPU cluster
- Tracks (or other physics objects like jets) are reconstructed on demand



- Flexible design
  + Expandible, easier to balance
  - Requires more communication and software development (e.g. a la HPX)

HPX

```
std::vector<Host::Quadruplet>
CUDACellularAutomaton::run(Host::Event event)
{
    // hpx::lcos::local::channel<unsigned int> resourceQueue;
    auto f_bufferIndex = resourceQueue.get();
    // May suspend if no buffer available
    const unsigned int bufferIndex = f_bufferIndex.get();

    copyEventToPinnedBuffers(event, bufferIndex);

    /* Same thing for streams... */

    // No HPX calls beyond this point, to avoid suspending
    cudaSetDevice(gpuIndex);
    asyncCopyEventToGPU(bufferIndex, streamIndex);

    /* ... */
}
```

```
std::vector<Host::Quadruplet>
CUDACellularAutomaton::run(Host::Event event)
{
    /* ... */

    // Define grid and block dimensions
    const std::array<unsigned int, 3> blockSize{256, 1, 1};
    const std::array<unsigned int, 3> numberOfBlocks_create{
        32, h_events[bufferIndex].numberOfLayerPairs, 1};

    // Call kernels through C++ wrappers
    kernel::create(
        numberOfBlocks_create, blockSize,
        0, streams[streamIndex],
        /* device pointers */);

    /* ... */
}
```

HPX

```
std::vector<Host::Quadruplet>
CUDACellularAutomaton::run(Host::Event event)
{
    /* ... */

    // Create quadruplet vector
    auto quadruplets = makeQuadrupletVector(bufferIndex);

    // Return resources, so other threads can use them
    resourceQueue.set(bufferIndex);
    streamQueue.set(streamIndex);

    // Return result
    return quadruplets;
}
```

- Wait for the results in another HPX thread, launched using hpx::async before even starting to send the next batch.
- Parallelize the main loop (one thread per CA worker) using hpx::parallel::for_loop. This automatically takes care of load-balancing.
- Send batches of events from several threads to each worker, to keep them constantly busy.

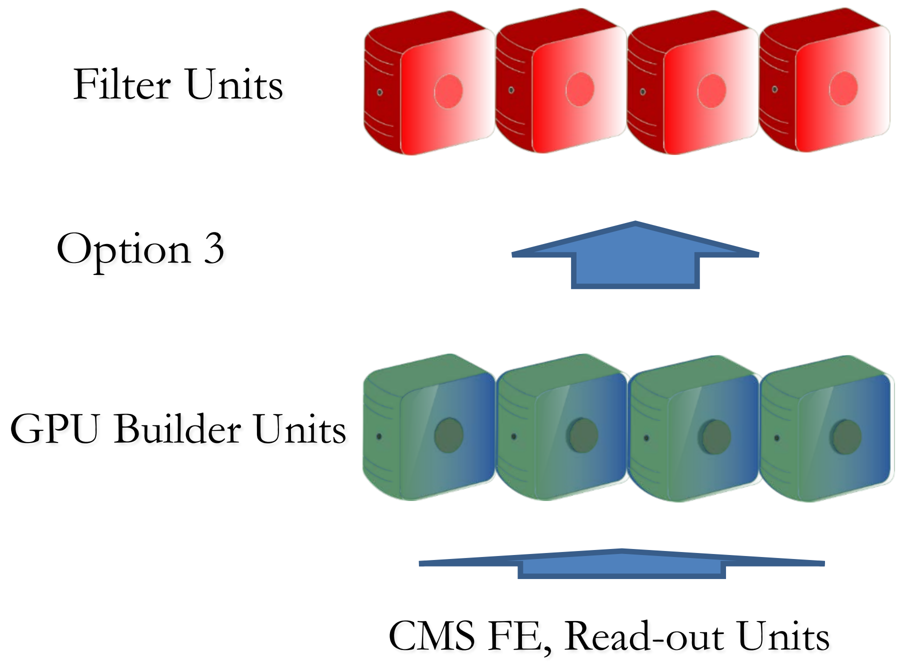
HPX

```
while (idx < nEvents)
{
    const std::size_t nextBatchIdx =
        std::min(idx + batchSize, nEvents);
    // Send futures
    for (std::size_t i = idx ; i < nextBatchIdx ; ++i) {
        const auto &ca = cellularAutomatons[i % nGPUs];
        const auto &evt = events[i % nEvents];
        f_allQuadruplets[i] = hpx::async(ca_action, ca, evt);
    }

    // Wait for the results in-order
    for (std::size_t i = idx ; i < nextBatchIdx ; ++i) {
        allQuadruplets[i] = f_allQuadruplets[i].get().size();
    }

    idx = nextBatchIdx;
}
```

- Wait for the results in another HPX thread, launched using hpx::async before even starting to send the next batch.
- Parallelize the main loop (one thread per CA worker) using hpx::parallel::for_loop. This automatically takes care of load-balancing.
- Send batches of events from several threads to each worker, to keep them constantly busy.

Integration in the HLT Farm

- Builder units are equipped with GPUs:
  - events with already reconstructed tracks are fed to FUs with GPUDirect
  - Use the GPU DRAM in place of ramdisks for building events.



- Very specific design

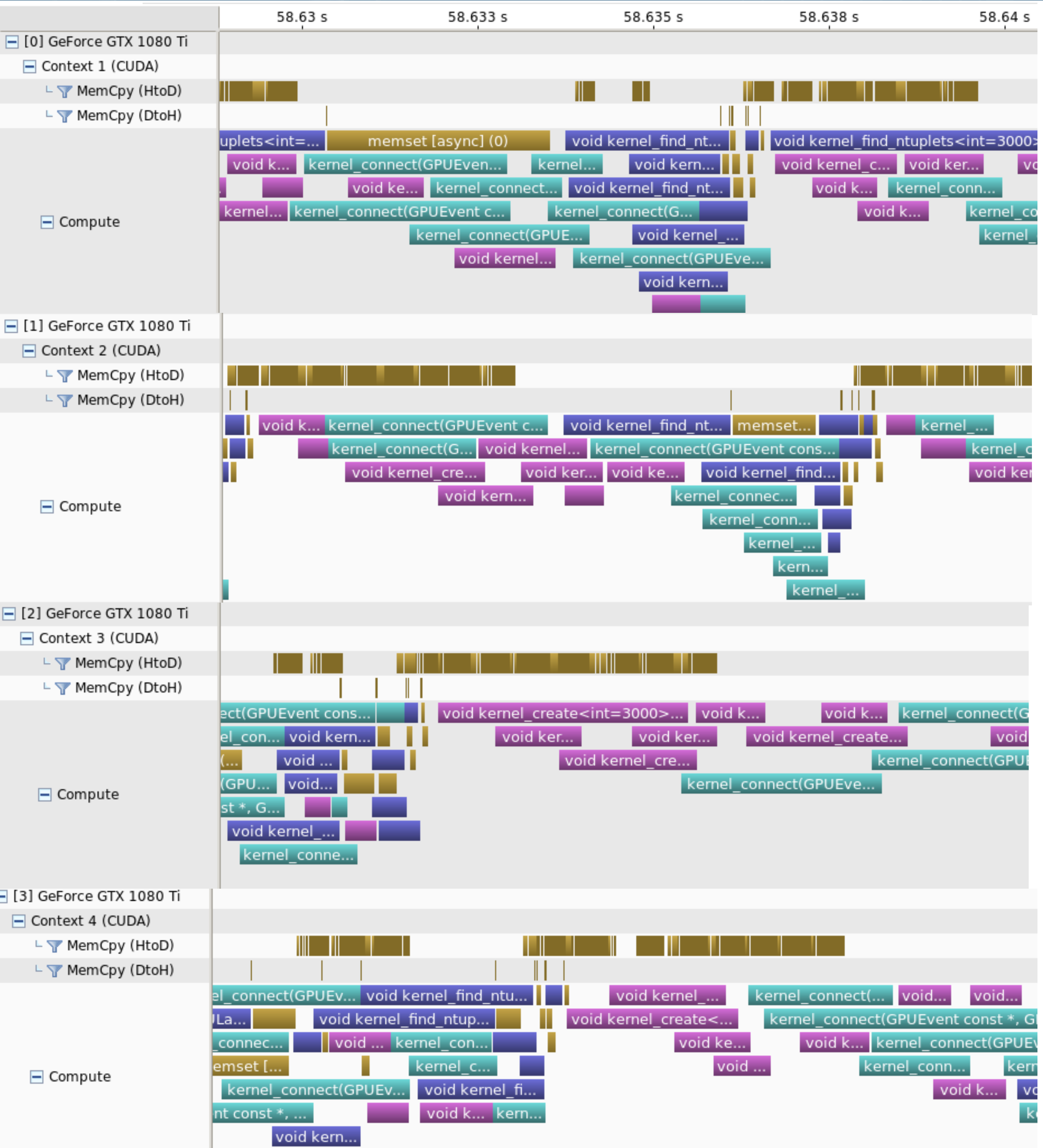
  + fast, independent of FU developments, integrated in readout

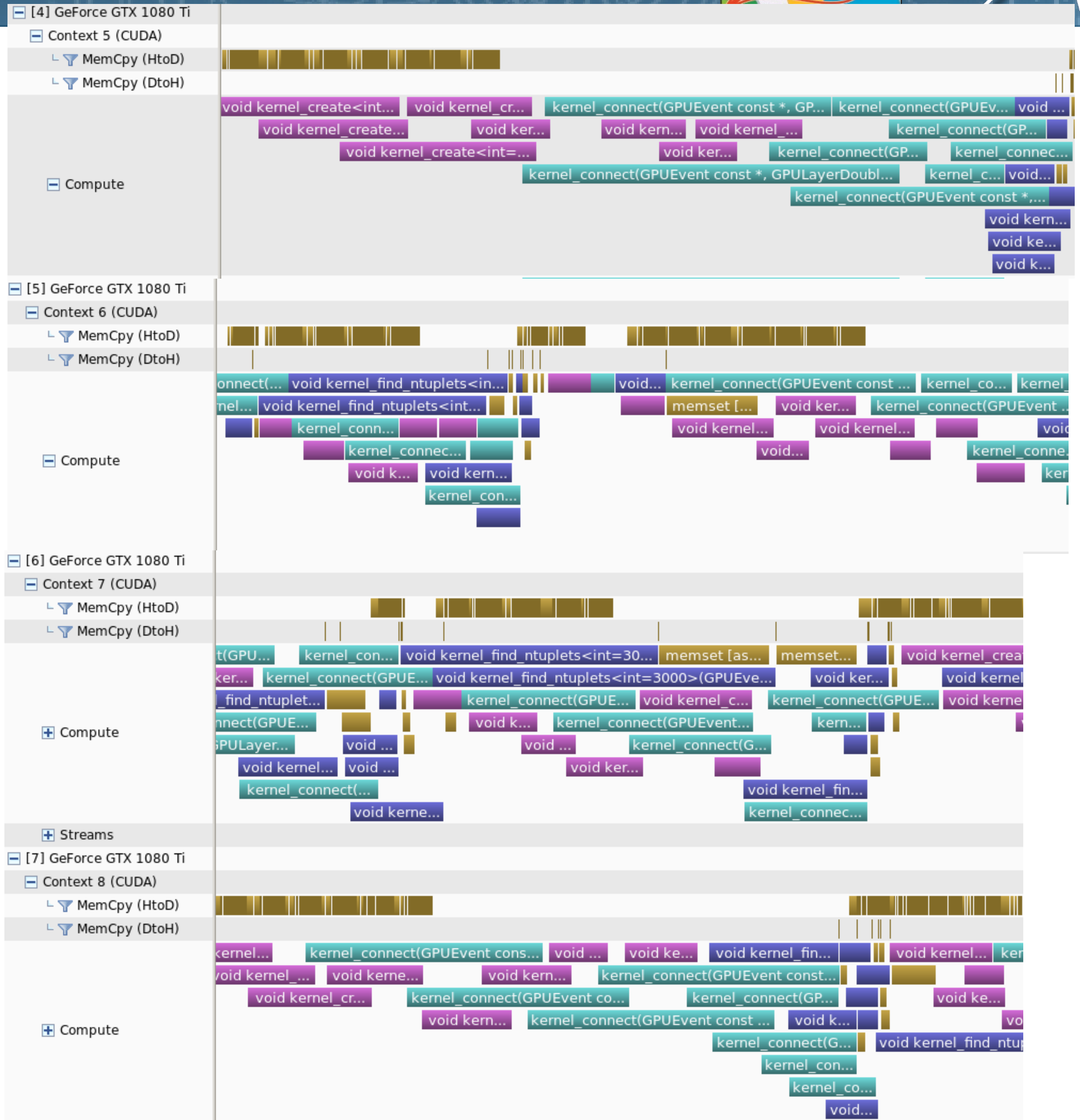
  - Requires specific DAQ software development: GPU may be “seen” as a detector element

Rate test

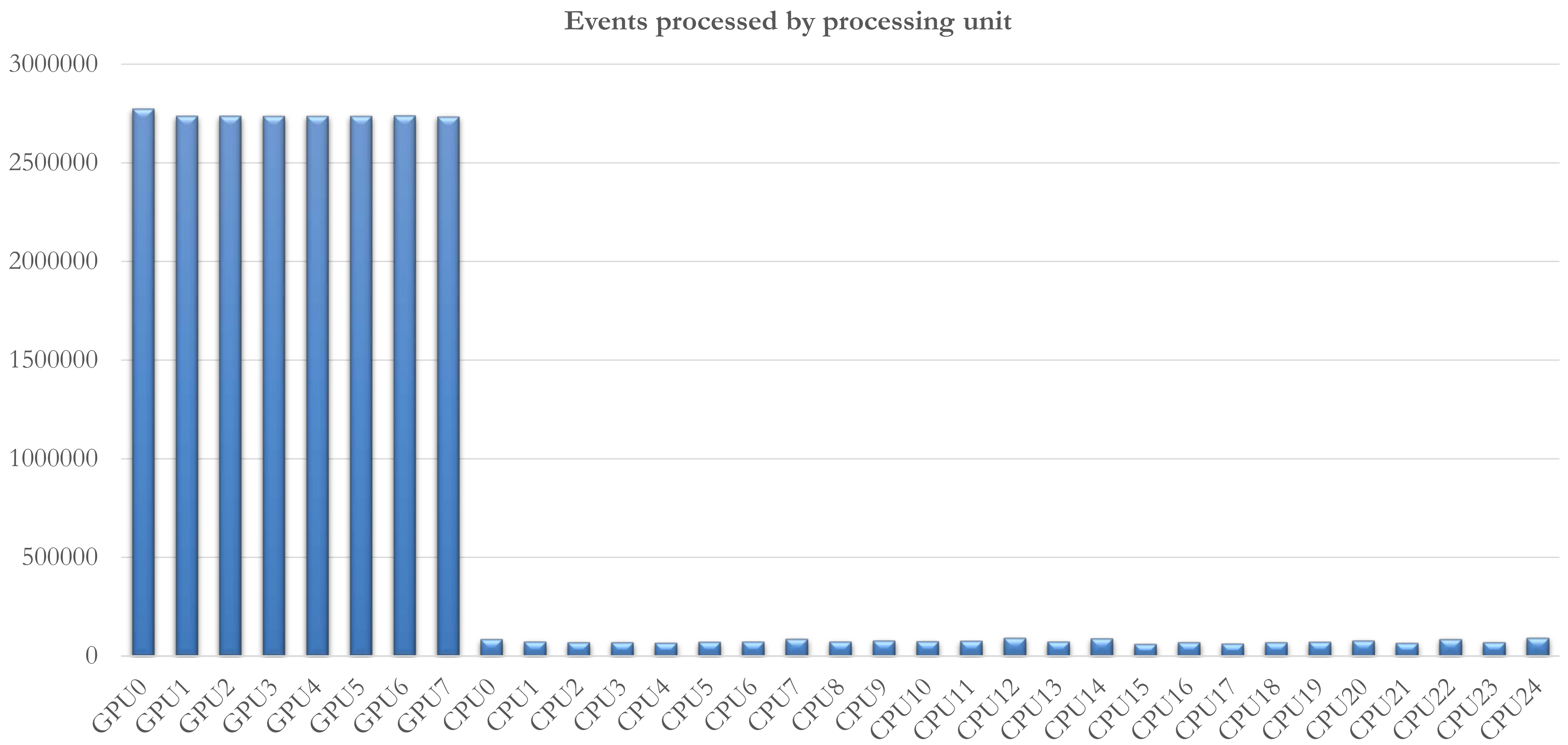
- The rate test consists in:
  - preloading in host memory few hundreds events
  - Assigning a host thread to a host core
  - Assigning a host thread to a GPU
  - Preallocating memory for each GPU for each of 8 cuda streams
  - Filling a concurrent queue with event indices
  - During the test, when a thread is idle it tries to pop from the queue a new event index:
    - Data for that event are copied to the GPU (if the thread is associated to a GPU)
    - processes the event (exactly same code executing on GPUs and CPUs)
    - Copy back the result
  - The test ran for approximately one hour
  - At the end of the test the number of processed events per thread is measured, and the total rate can be estimated

What happens in 10ms



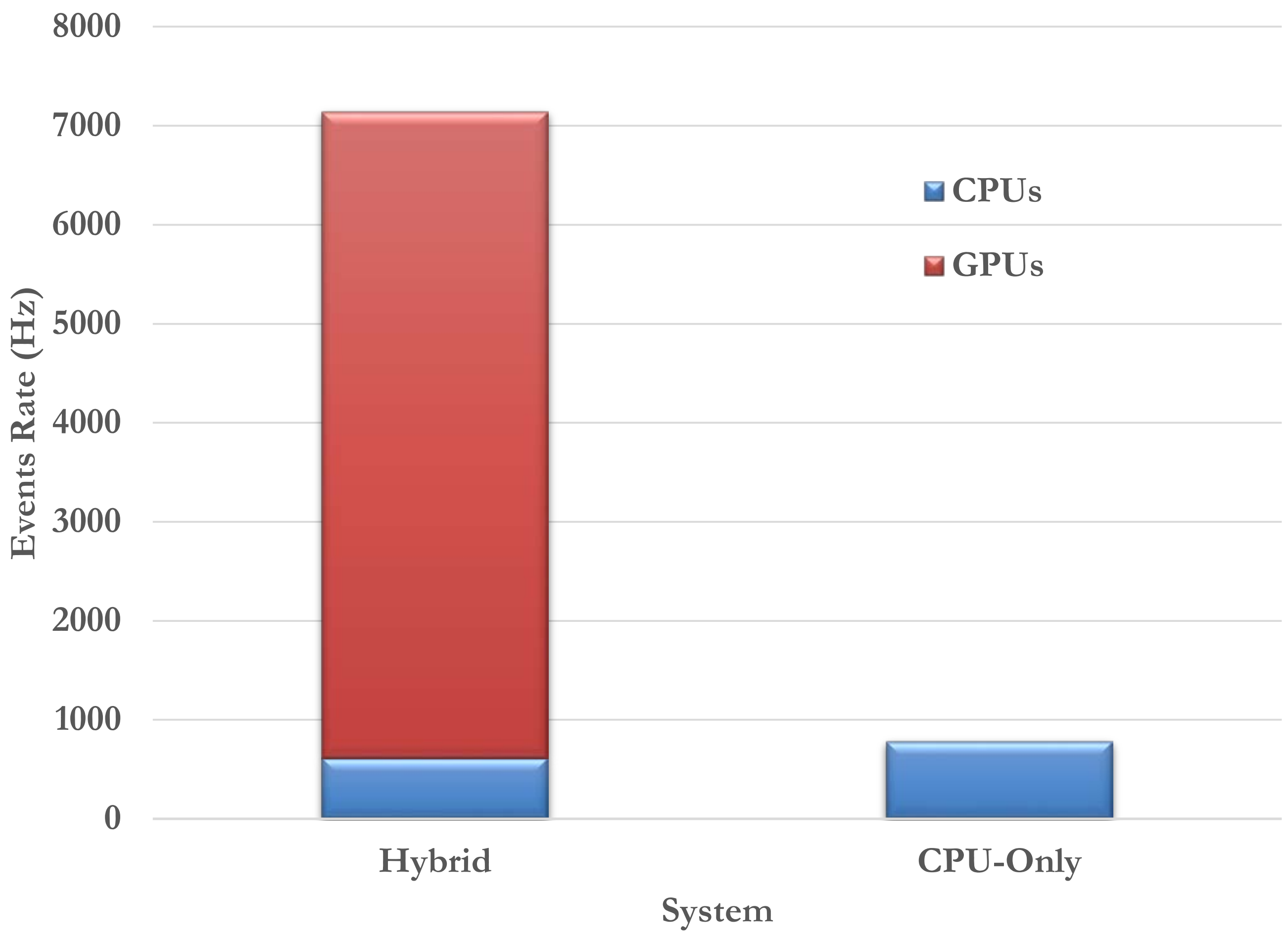


Rate test


Events processed by processing unit
3000000
2500000
2000000
1500000
1000000
500000
0
GPU0
GPU1
GPU2
GPU3
GPU4
GPU5
GPU6
GPU7
CPU0
CPU1
CPU2
CPU3
CPU4
CPU5
CPU6
CPU7
CPU8
CPU9
CPU10
CPU11
CPU12
CPU13
CPU14
CPU15
CPU16
CPU17
CPU18
CPU19
CPU20
CPU21
CPU22
CPU23
CPU24

Rate test

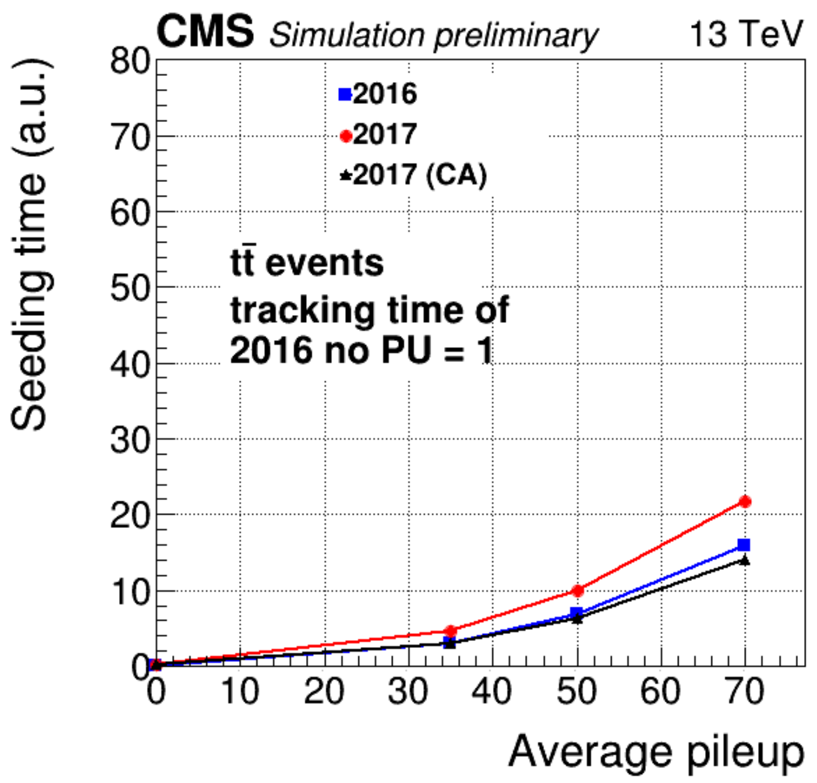
- CERN acquired a small machine for development and testing
  - Special configuration
- CPU-only input rate:
  - Rate with 24xCPUs: 777 Hz
  - Number of nodes to reach 100kHz: ~128
  - 4 Events per Joule
- Hybrid input rate:
  - 8xGPU: 6527 Hz + 24xCPUs: 613 Hz
  - Number of nodes to reach 100kHz: ~14
  - 6.x + 3.2 Events per Joule



2 sockets x Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz (2x12 physical cores)
256GB system memory
8x GPUs NVIDIA GTX 1080Ti

Timing vs PU

- CA track seeding at same level of the 2016 seeding
- More robust, smaller complexity vs PU than 2016 track seeding despite the increased number of layer combinations involved in the seeding phase with respect to the 2016 seeding
- ~25% faster track reconstruction wrt to 2016 tracking at avg PU70
- Replacing the CMS Phase2 offline track seeding with sequential CA
  - Overall tracking 2x faster at PU200
  - T(Phase2Tracker@PU200) = 4xT(Phase1Tracker@PU50)
    - Detector and algorithms defeated combinatorial complexity



Conclusion

Conclusion

- The future runs of the Large Hadron Collider at CERN will impose significant challenge on the software performance, due to the increasing complexity of events
- This pioneering work presents ways to solve the compute intensive problem of track seeding in the CMS Pixel Detector:
  - Hit-Chain Maker improves physics performance while being significantly faster than the existing sequential implementation
  - Started porting of other parts of the reconstruction: the heterogeneous revolution has begun!
- It has replaced the existing track seeding algorithm starting from the 2017 data-taking both in the Online and Offline event reconstruction
- Complete demonstrator of the GPU-based Pixel Tracking to be installed in Autumn 2018 at the CMS High-Level Trigger farm at the LHC Point 5
- Final target: LHC Run 3

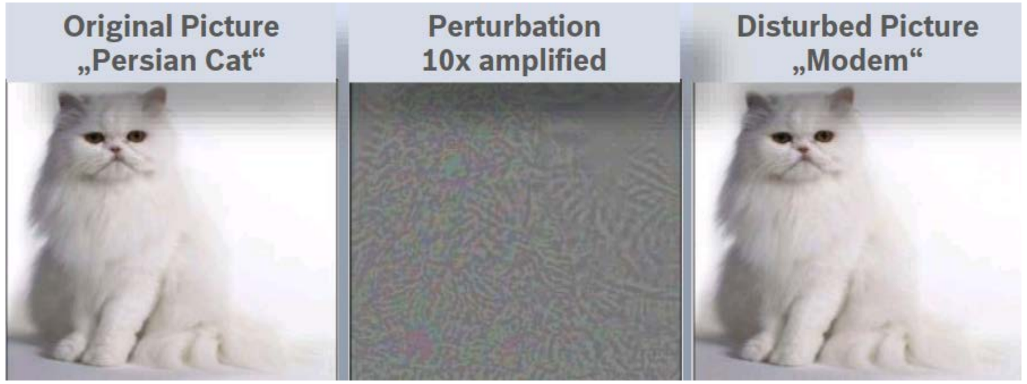
That's all Folks!

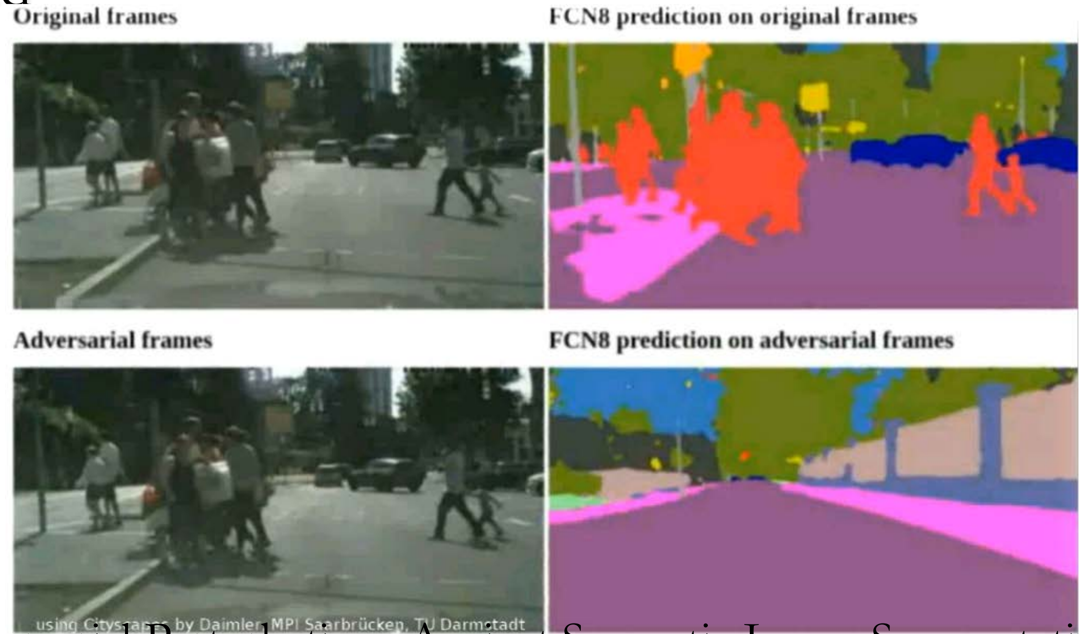
Fun Fact

- What is the difference between a cat and a modem?

Fun fact: Adversarial Threats

- What is the difference between a cat and a modem?
- I apologize to any AI listening to this talk, this kind of humor is simply not acceptable..
- Research on unsupervised learning required


Original Picture „Persian Cat"
Perturbation 10x amplified
Disturbed Picture „Modem"


Original frames
FCN8 prediction on original frames
Adversarial frames
FCN8 prediction on adversarial frames
using Cityscapes by Daimler, MPI Saarbrücken, TU Darmstadt

J. H. Metzen, K. C. Mummadi, T. Brox, V. Fischer, "Universal Adversarial Perturbations Against Semantic Image Segmentation", The IEEE International Conference on Computer Vision (ICCV), 2017

PATATRACK

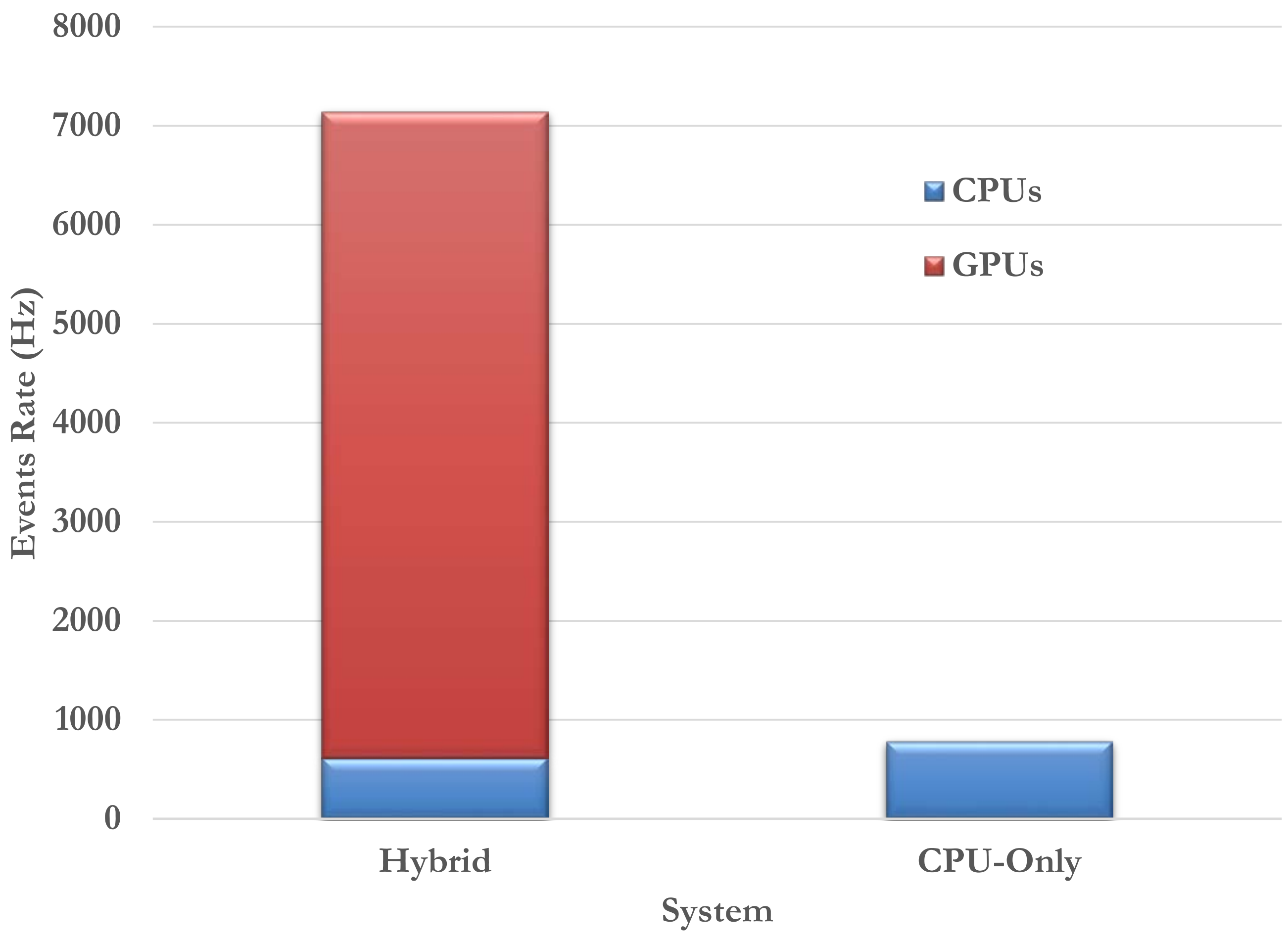
Started in 2016 by a very small group of passionate people, right after I gave a GPU programming course…

- Soon grown:
  - CERN: F. Pantaleo, V. Innocente, M. Rovere, A. Bocci, M. Kortelainen, M. Pierini, V. Volkl (SFT), V. Khristenko (IT, openlab)
  - Austrian Academy of Sciences: E. Brondolin, R. Fruhwirth
  - INFN Bari: A. Di Florio, C. Calabria
  - INFN MiB: D. Menasce, S. Di Guida
  - INFN CNAF: E. Corni
  - SAHA: S. Sarkar, S. Dutta, S. Roy Chowdhury, P. Mal
  - TIFR: S. Dugad, S. Dubey
  - Aachen: A. Schmidt
  - University of Pisa (Computer Science dep.): D. Bacciu, A. Carta
  - Thanks also to the contributions of many short term students (Bachelor, Master, GSoC): Alessandro, Ann-Christine, Antonio, Dominik, Jean-Loup, Konstantinos, Kunal, Luca, Panos, Roberto, Romina, Simone, Somesh
- Interests: algorithms, HPC, heterogeneous computing, machine learning, software eng., FPGAs…
- Lay the foundations of the online/offline reconstruction starting from 2020s (tracking, HGCal)
- Website under construction: PATATRACK , contact: patatrack-rd@cern.ch

Rate test

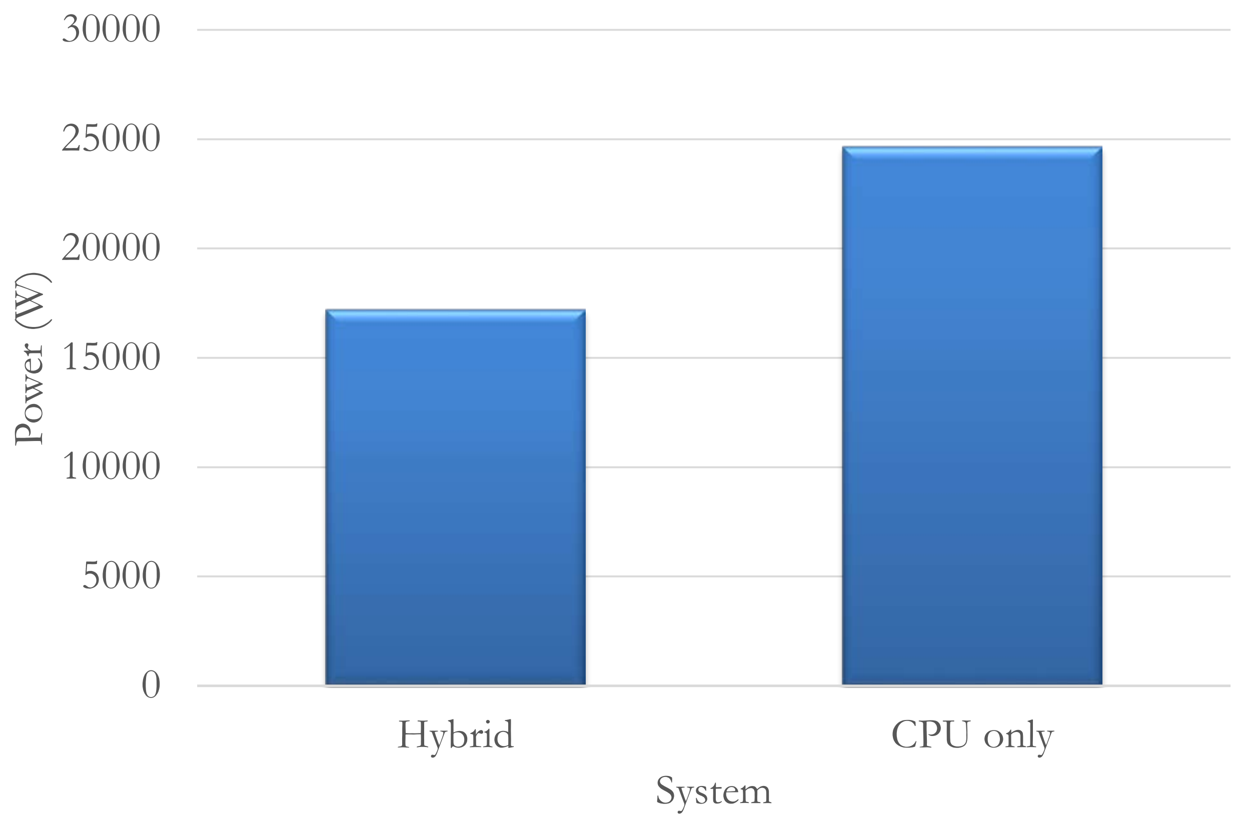
- Total rate measured:
  - 8xGPU: 6527 Hz
  - 24xCPUs: 613 Hz
- Number of nodes to reach 100kHz: ~14
- Total Price: 70x 🍌

- When running with only 24xCPUs
  - Rate with 24xCPUs: 777 Hz
- Number of nodes to reach 100kHz: ~128
- Total Price: 320x 🍌
  - Assuming an initial cost of 2.5 per node



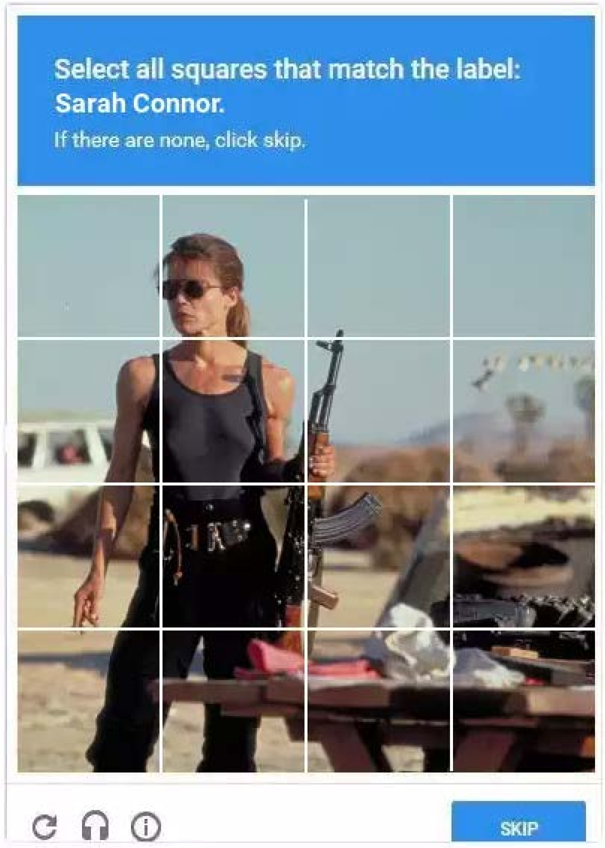
Energy efficiency

- During the rate test power dissipated by CPUs and GPUs was measured every second
  - Nvidia-smi for GPUs
  - Turbostat for CPUs
- 8 GPUs: 1037W
  - 6.29 Events per Joule
  - 0.78 Events per Joule per GPU
- 24 CPUs in hybrid mode: 191W
  - 3.2 Events per Joule
  - 0.13 Events per Joule per core
- 24 CPUs in CPU-only test: 191W
  - 4.05 Events per Joule
  - 0.17 Events per Joule per core
- That is 1/3 more 🍌s in the energy bill when processing 100kHz input



Conclusion

- Hardware is changing
  - Yes, again
- Today heterogeneous computing is not the exception
  - Data centers are assumed to host several different kinds of accelerators
- Duck test:
  - Pixel Track seeding algorithms have been redesigned with high-throughput parallel architectures in mind
  - This is only the beginning…
  - … many other parts of the HLT menu will be targeted to become heterogeneous
- Improvements in performance may come even when running sequentially
  - Factors at the HLT, tens of % in the offline, depending on the fraction of the code that use new algos
- The GPU and CPU algorithms run in CMSSW and produce the same result
  - Transition to GPUs@HLT during Run3 smoother
- Complete demonstrator will be installed at Point5 in Autumn 2018 to run with real data


Select all squares that match the label:
Sarah Connor.
If there are none, click skip.

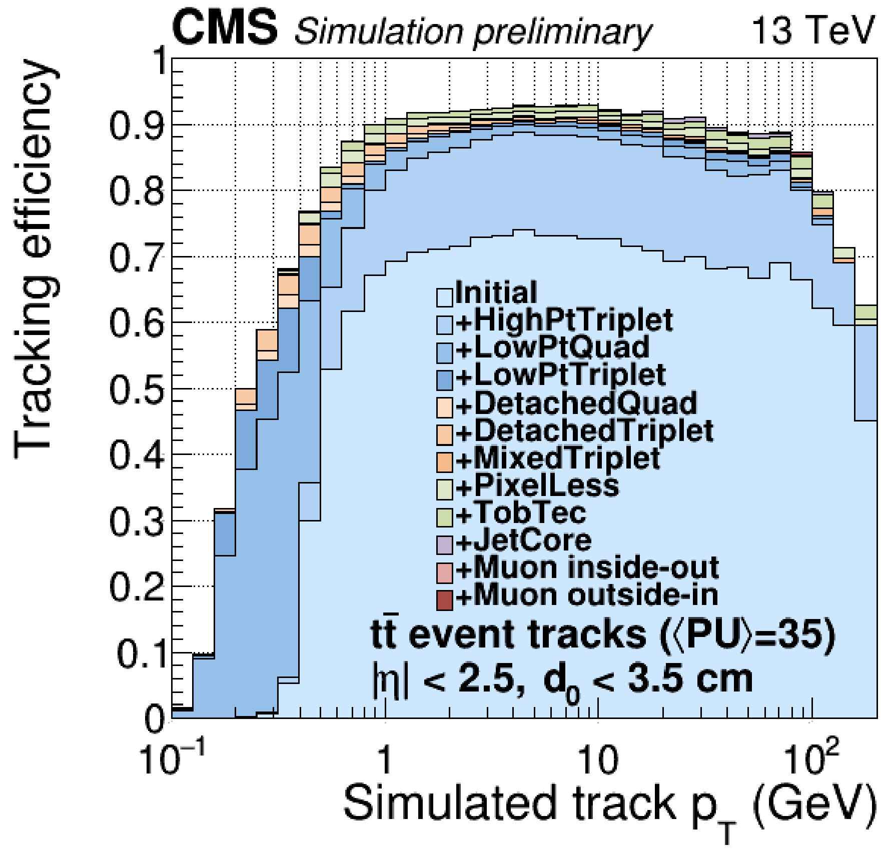
Back up

CA-based Hit Chain Maker@ Run-2 Offline Track Seeding

CA in offline tracking

- The performance of the sequential Cellular Automaton at the HLT justified its integration also in the 2017 offline iterative tracking

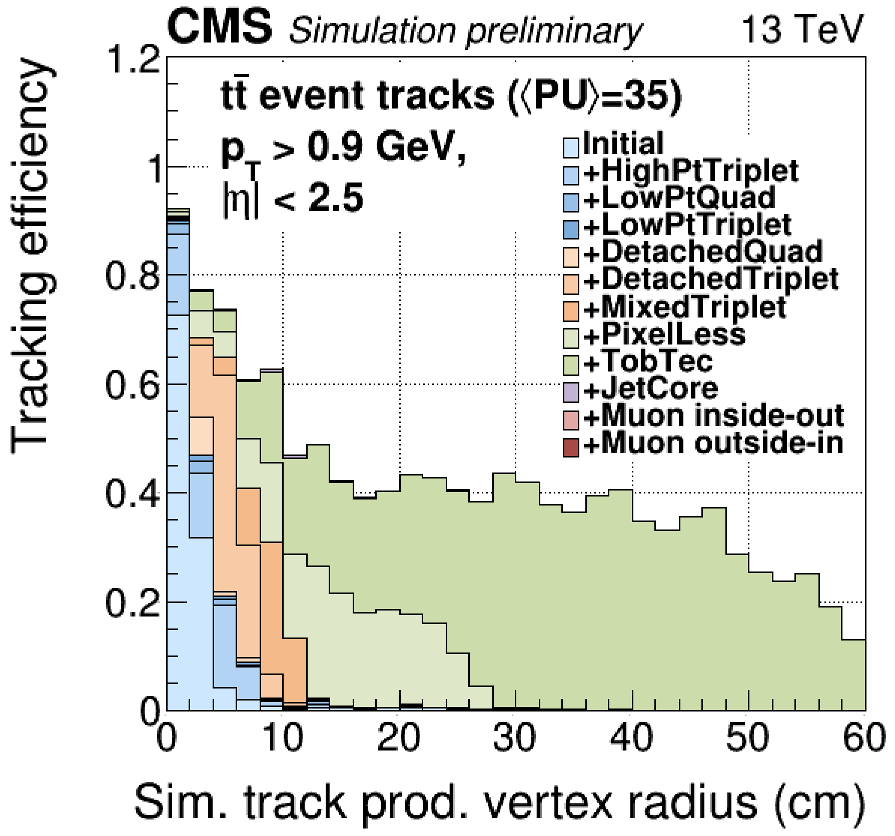
Iteration	Seeding	Target track
Initial	pixel quadruplets	prompt, high p_T
LowPtQuad	pixel quadruplets	prompt, low p_T
HighPtTriplet	pixel triplets	prompt, high p_T recovery
LowPtTriplet	pixel triplets	prompt, low p_T recovery
DetachedQuad	pixel quadruplets	displaced--
DetachedTriplet	pixel triplets	displaced-- recovery
MixedTriplet	pixel+strip triplets	displaced-
PixelLess	inner strip triplets	displaced+
TobTec	outer strip triplets	displaced++
JetCore	pixel pairs in jets	high-p_T jets
Muon inside-out	muon-tagged tracks	muon
Muon outside-in	standalone muon	muon



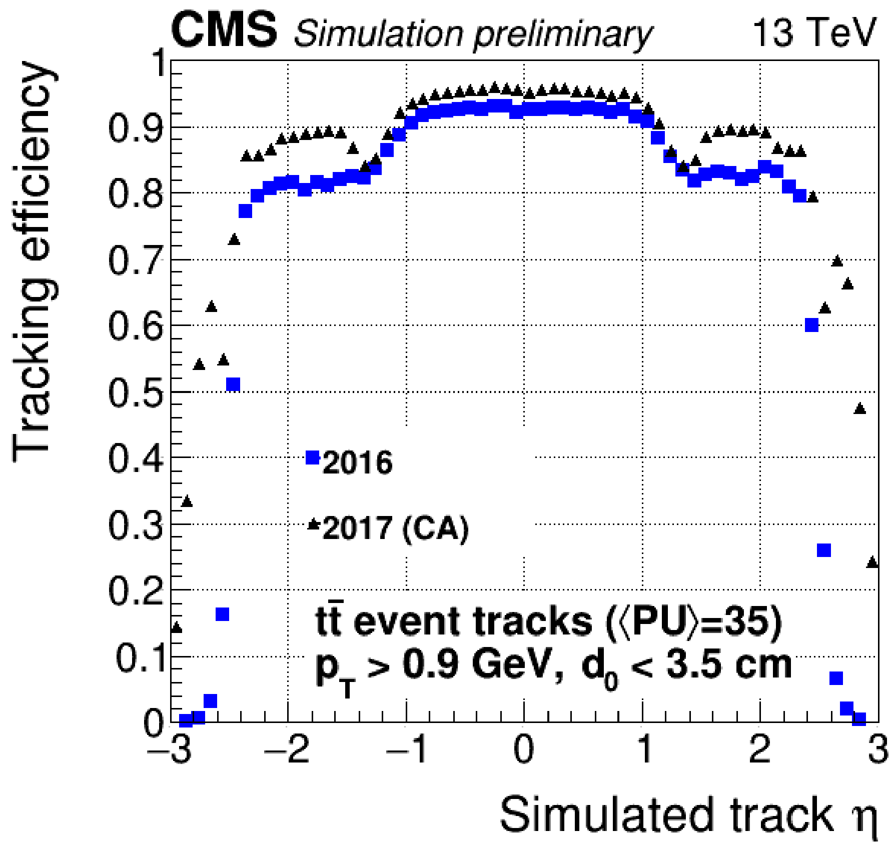
CA in offline tracking

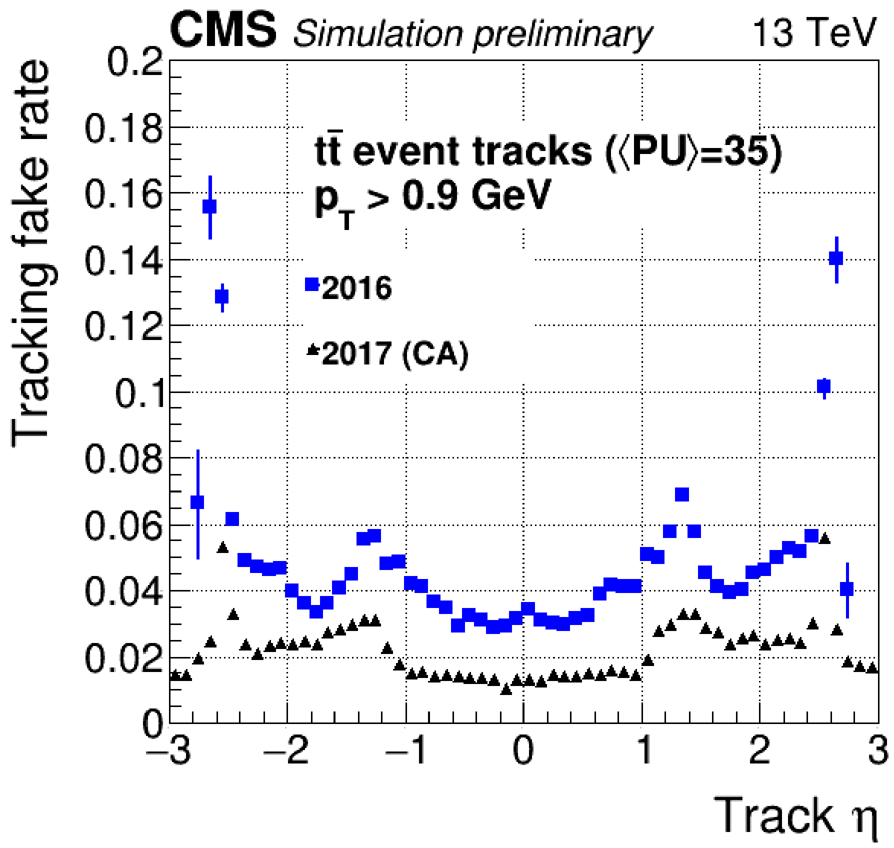
- The performance of the sequential Cellular Automaton at the HLT justified its integration also in the 2017 offline iterative tracking

Iteration	Seeding	Target track
Initial	pixel quadruplets	prompt, high p_T
LowPtQuad	pixel quadruplets	prompt, low p_T
HighPtTriplet	pixel triplets	prompt, high p_T recovery
LowPtTriplet	pixel triplets	prompt, low p_T recovery
DetachedQuad	pixel quadruplets	displaced--
DetachedTriplet	pixel triplets	displaced-- recovery
MixedTriplet	pixel+strip triplets	displaced-
PixelLess	inner strip triplets	displaced+
TobTec	outer strip triplets	displaced++
JetCore	pixel pairs in jets	high-p_T jets
Muon inside-out	muon-tagged tracks	muon
Muon outside-in	standalone muon	muon



CA Physics performance vs 2016

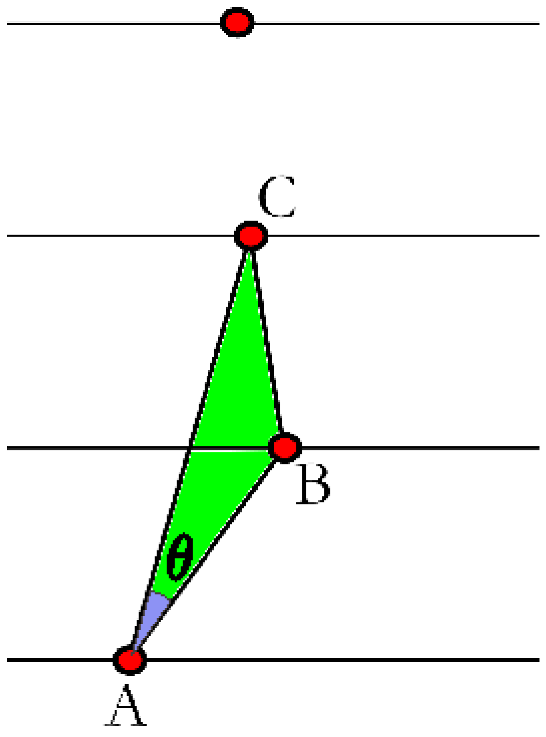




- Reconstruction efficiency increased
  - especially in forward region.
- Fake rate significantly reduced in the entire pseudo-rapidity range

CA: R-z plane compatibility

- The compatibility between two cells is checked only if they share one hit
  - AB and BC share hit B
- In the R-z plane a requirement is alignment of the two cells:
  - There is a maximum value of ϑ that depends on the minimum value of the momentum range that we would like to explore



CA: x-y plane compatibility

- In the transverse plane, the intersection between the circle passing through the hits forming the two cells and the beamspot is checked:
  - They intersect if the distance between the centers d(C,C') satisfies: $r' - r < d(C,C') < r' + r$
  - Since it is a Out – In propagation, a tolerance is added to the beamspot radius (in red)
- One could also ask for a minimum value of transverse momentum and reject low values of r'