New Track Seeding Techniques at the CMS experiment

Felice Pantaleo
CERN – EP-CMG

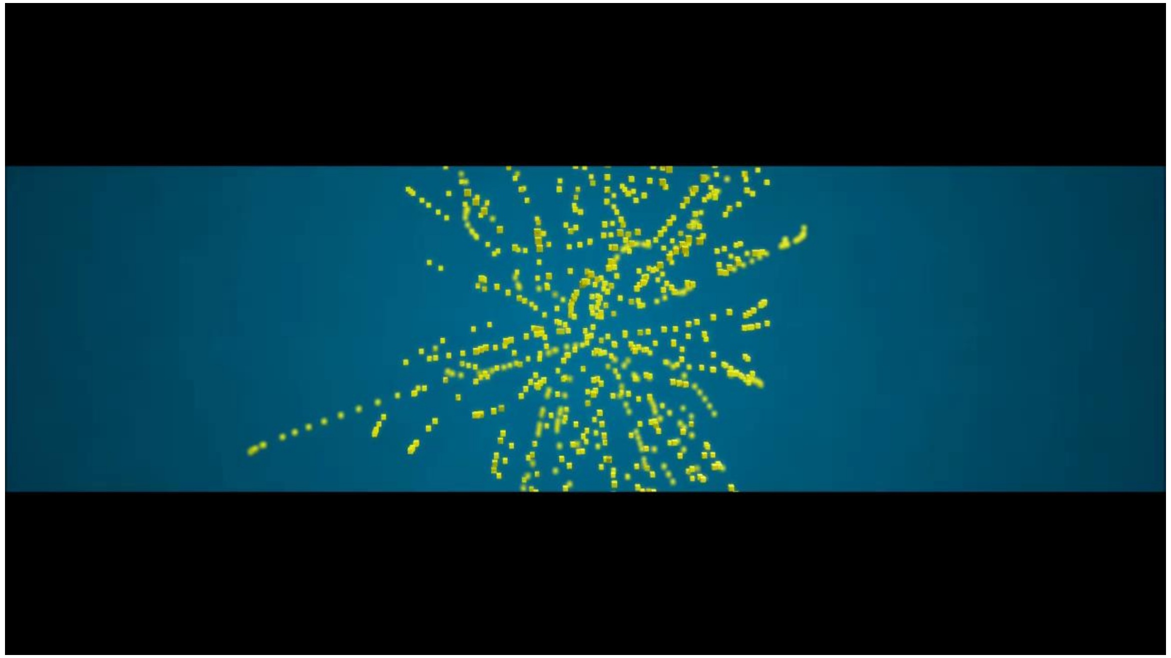
felice@cern.ch

Overview

- Motivations
- Track seeding on GPUs during Run-3
- Pixel Tracks today
  - Online
  - Offline
- Conclusion

Tracking

- Particles produced in the collisions leave traces (hits) as they fly through the detector
- The innermost detector of CMS is called **Silicon Tracker**
- **Tracking**: the art of reconstructing particles trajectories by connecting correctly hits together
- In a solenoidal magnetic field, trajectories are helices



Two-stages event selection strategy

Trigger System

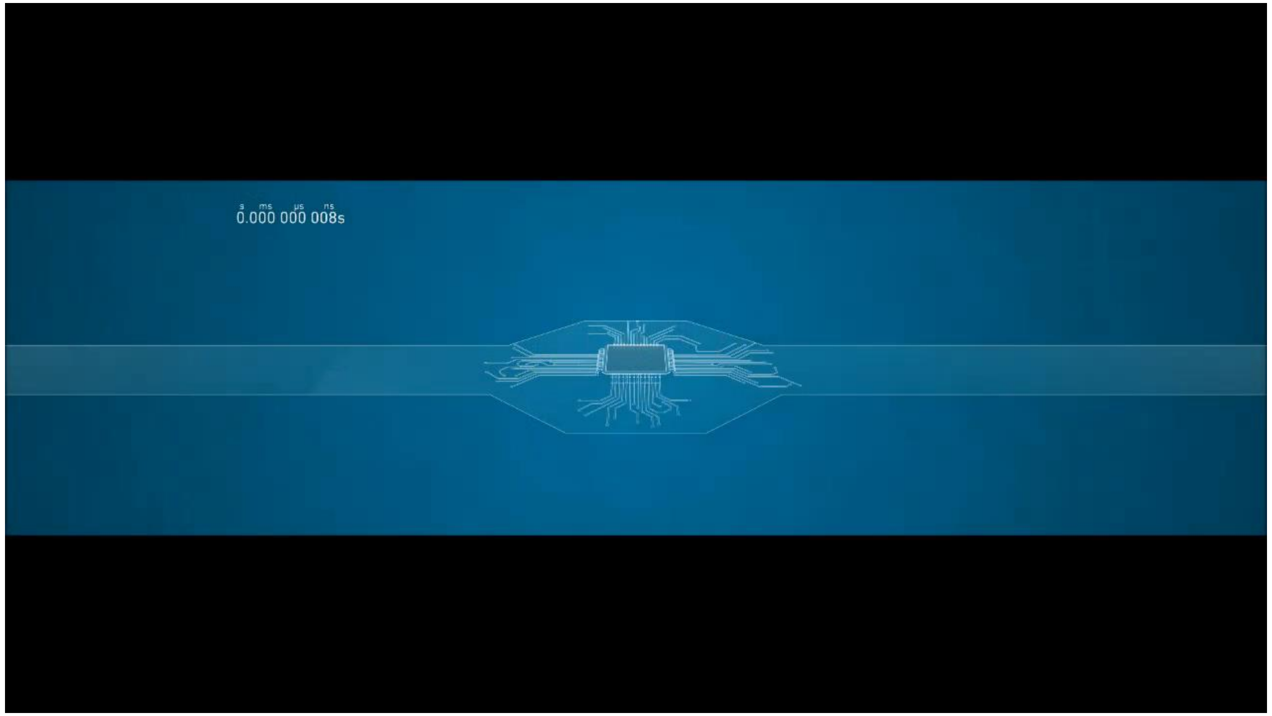
- Reduce input rate (**40 MHz**) to a data rate (**~1 kHz**) that can be stored, reconstructed and analyzed Offline maximizing the physics reach of the experiment

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

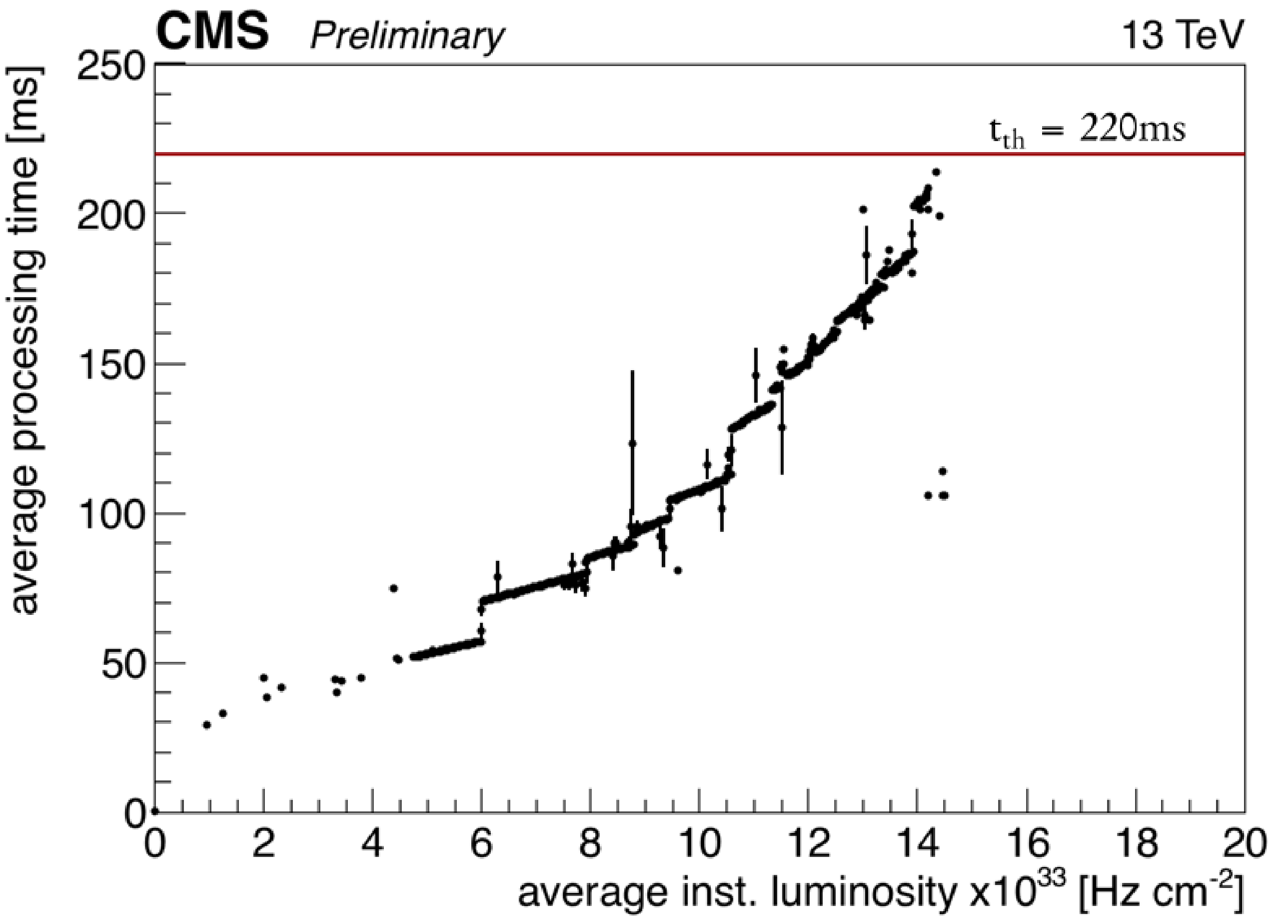
High Level Trigger

- readout of the whole detector with full granularity
- based on the CMSSW software, running on 22,000 Xeon cores
- organized in O(2500) modules, O(400) trigger paths, O(10) streams
- output rate limited to an average of **~1 kHz** by the Offline resources



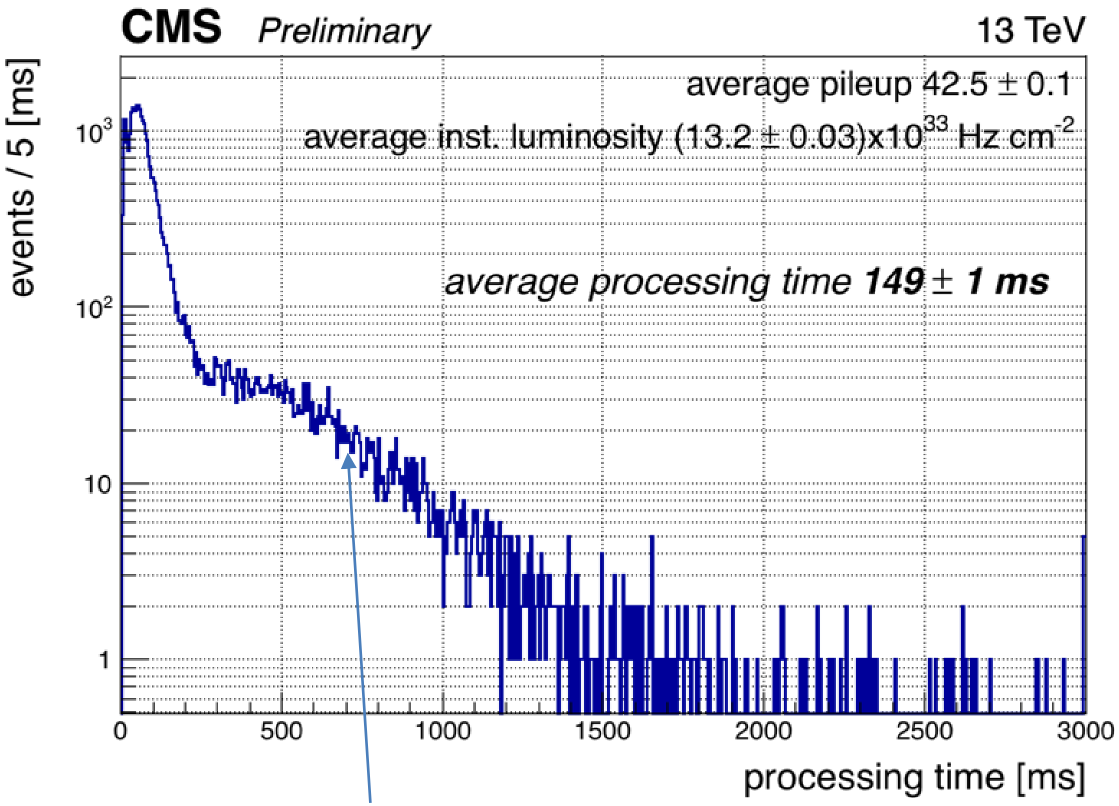
CMS High-Level Trigger in Run 2 (1/2)

- Today the CMS online farm consists of ~22k Intel Xeon cores
  - The current approach: one event per logical core
- Pixel Tracks are not reconstructed for all the events at the HLT
- This will be even more difficult at higher pile-up
  - More memory/event



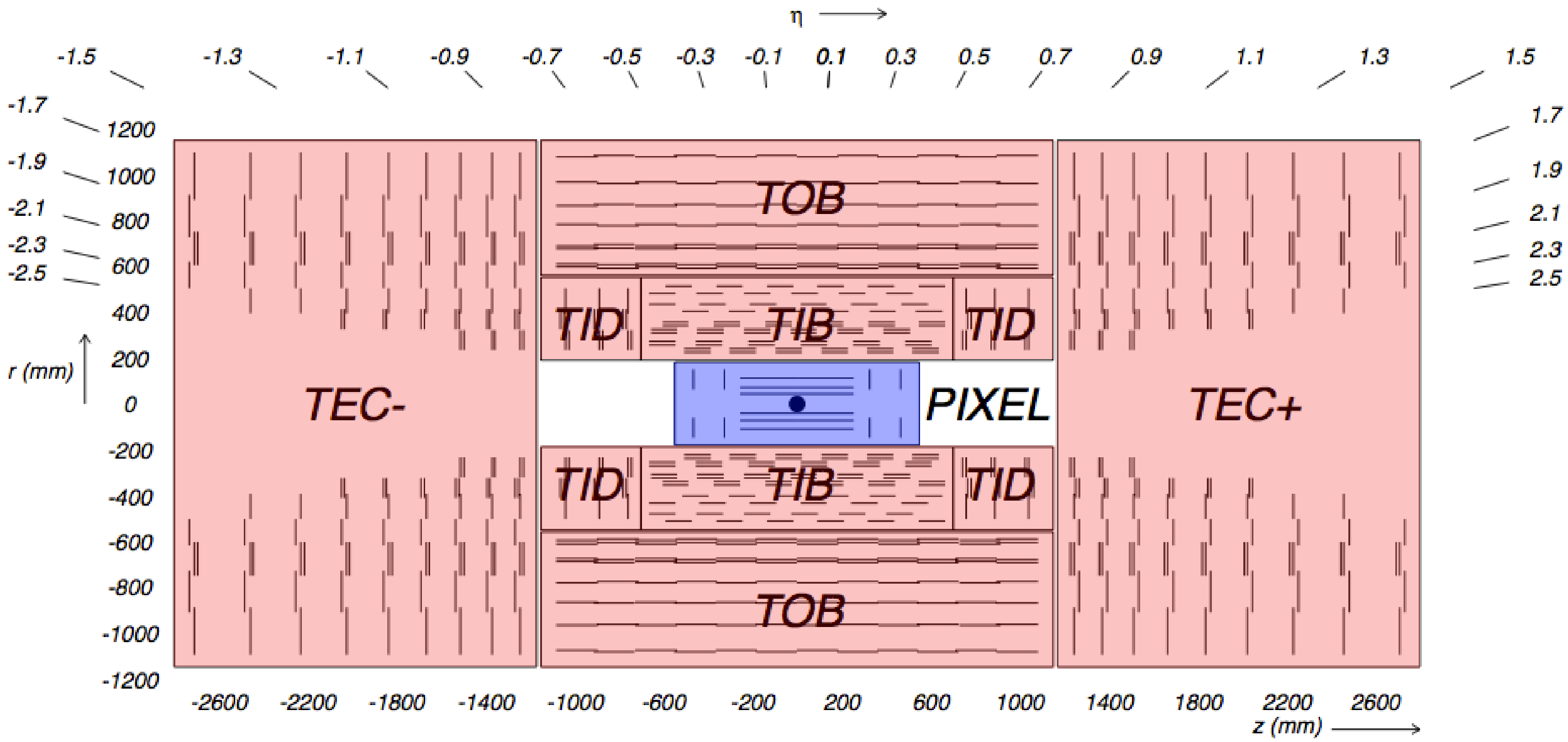
CMS High-Level Trigger in Run 2 (2/2)

- Today the CMS online farm consists of ~22k Intel Xeon cores
  - The current approach: one event per logical core
- Pixel Tracks are not reconstructed for all the events at the HLT
- This will be even more difficult at higher pile-up
  - More memory/event



full track reconstruction and particle flow e.g. jets, tau

Terminology - CMS Silicon Tracker



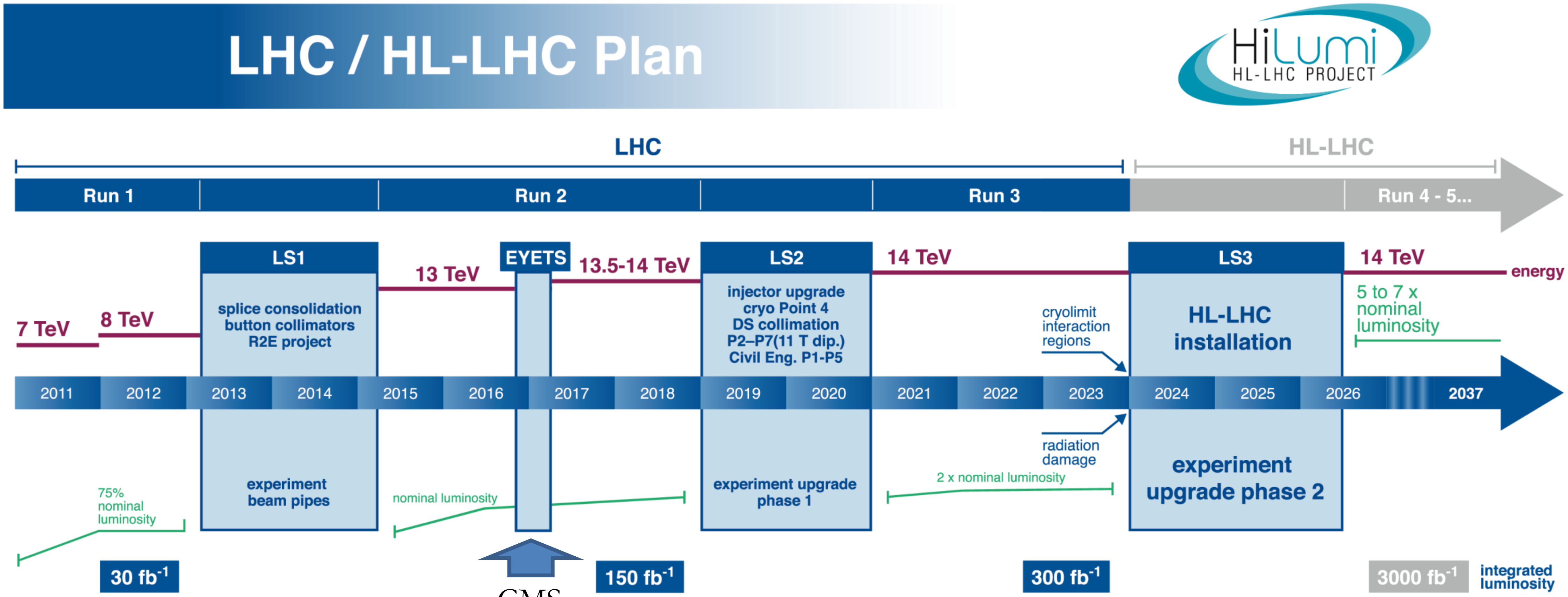
Online:

- Pixel-only tracks used for fast tracking and vertexing

Offline:

- Pixel tracks are used as seeds for the Kalman filter in the strip detector

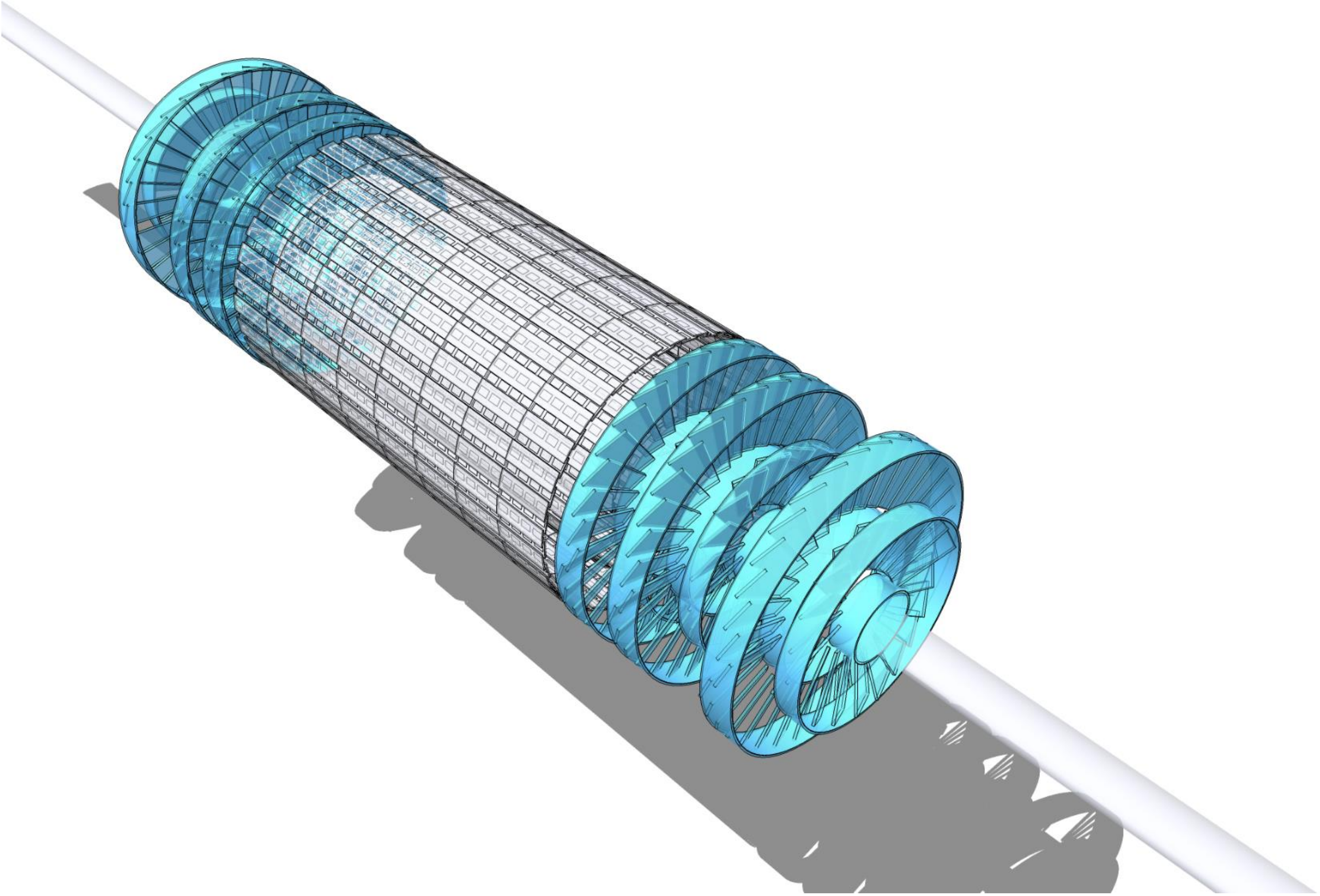
CMS and LHC Upgrade Schedule



CMS
pixel detector upgrade

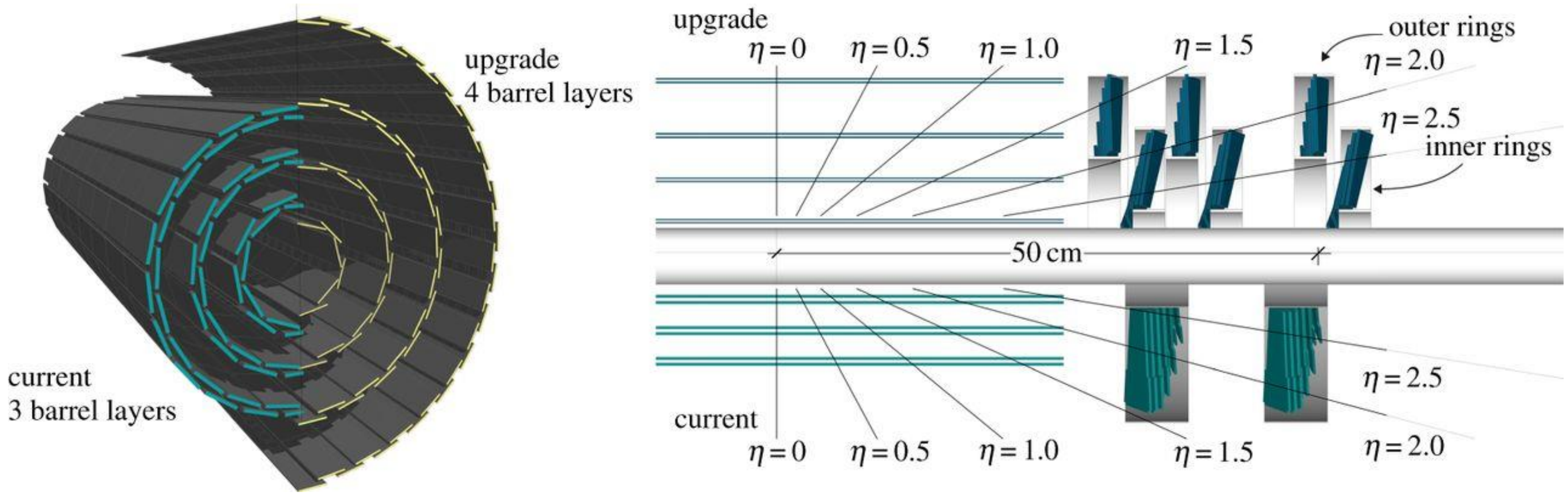
Phase 1 Pixel detector (1/2)

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.



Phase 1 Pixel detector (2/2)

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.



Tracking at HLT

- Pixel hits are used for pixel tracks, vertices, seeding
- HLT Iterative tracking:

Iteration name	Phase0 Seeds	Phase1 Seeds	Target Tracks
Pixel Tracks	triplets	quadruplets	
Iter0	Pixel Tracks	Pixel Tracks	Prompt, high p_T
Iter1	triplets	quadruplets	Prompt, low p_T
Iter2	doublets	triplets	High p_T, recovery

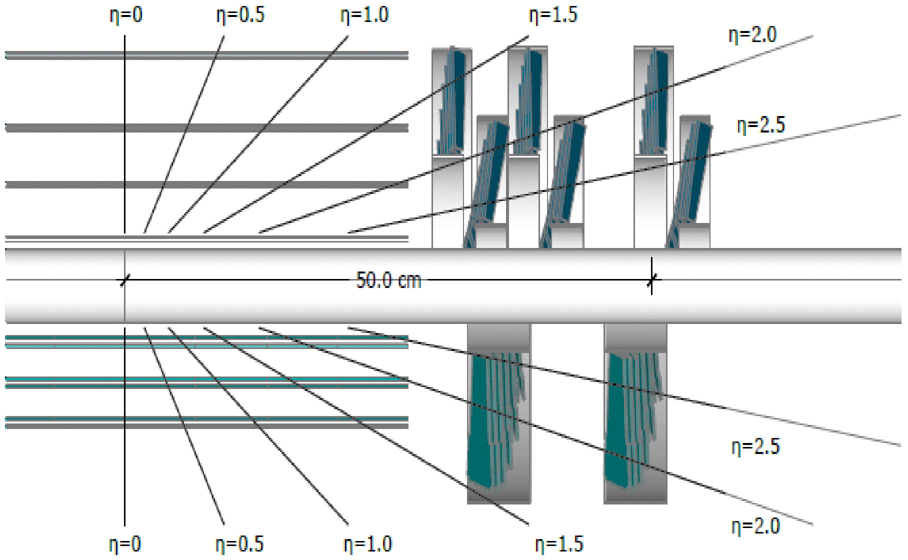
Pixel Tracks

- Evaluation of Pixel Tracks combinatorial complexity could easily be dominated by track density and become the bottleneck 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

Track seeding on GPUs during Run-3

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

↓

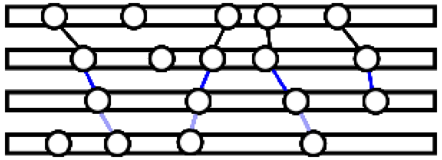
Hits - Pixel Clusterizer

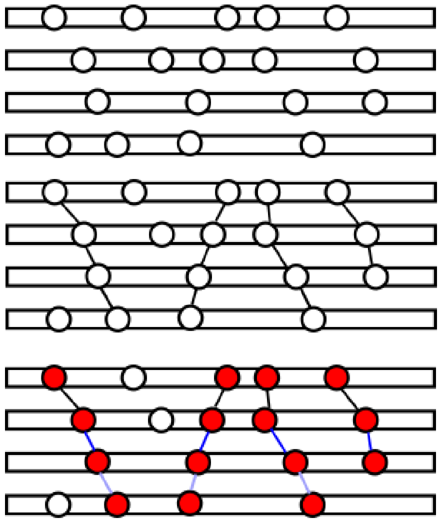
↓

Hit Pairs

↓

CA-based Hit Chain Maker



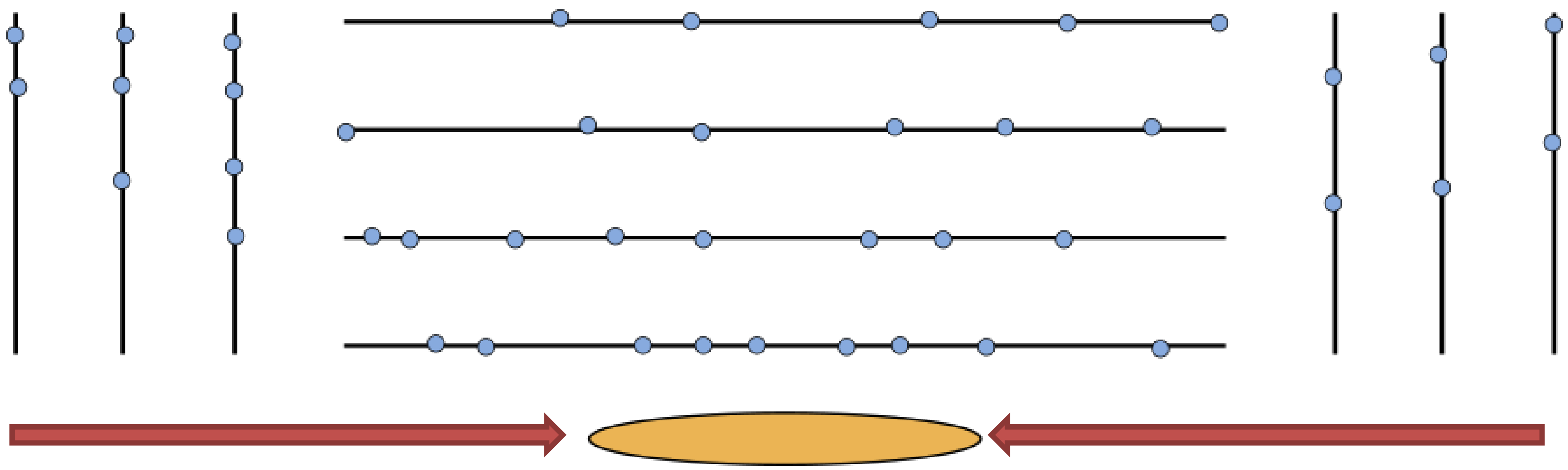


↓

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

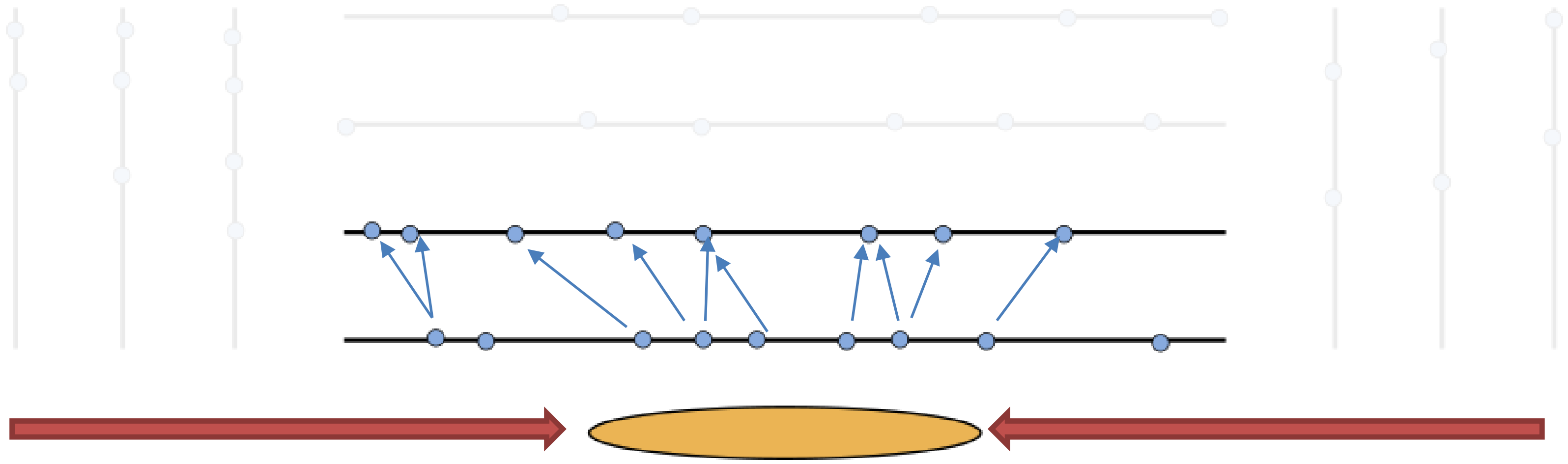
RMS HEP Algorithm

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



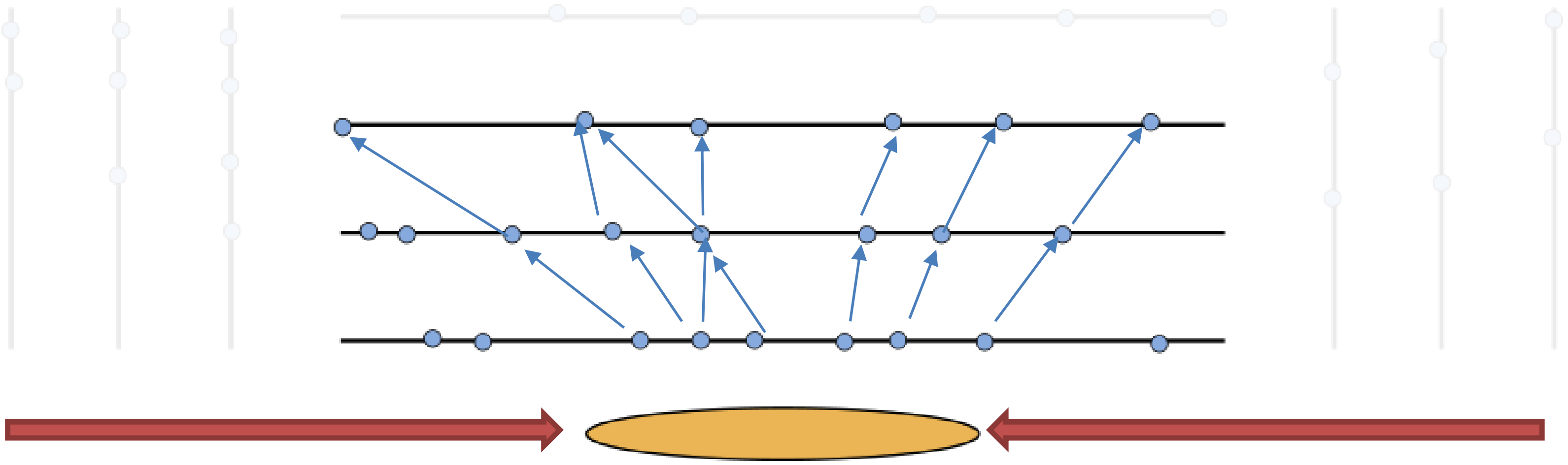
RMS HEP Algorithm

- First create doublets from hits of pairs



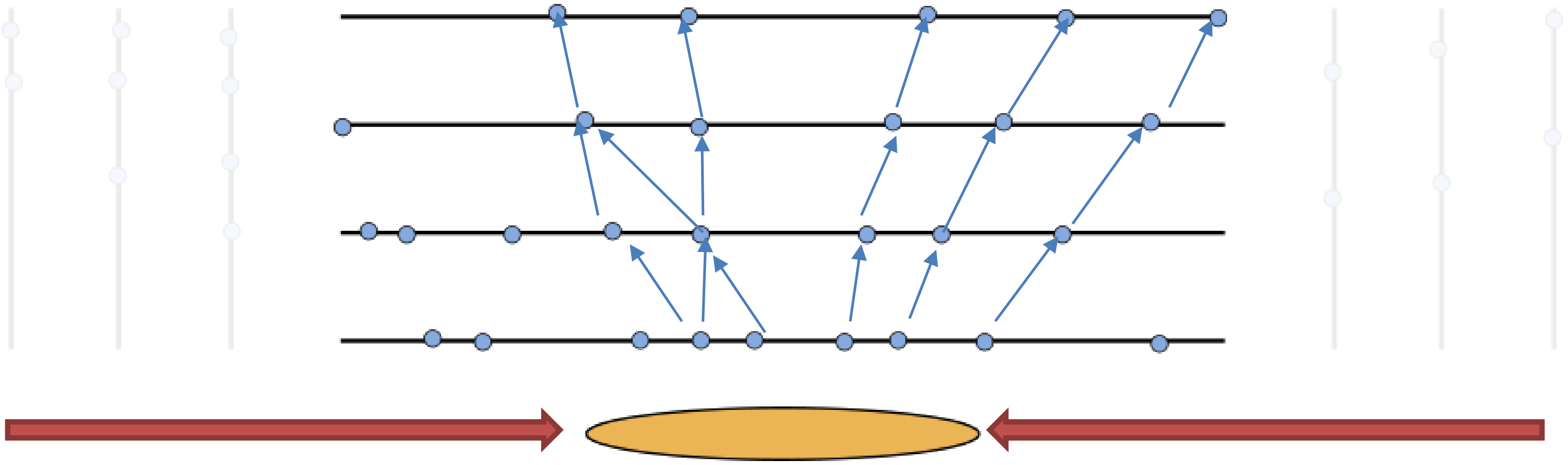
RMS HEP Algorithm

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



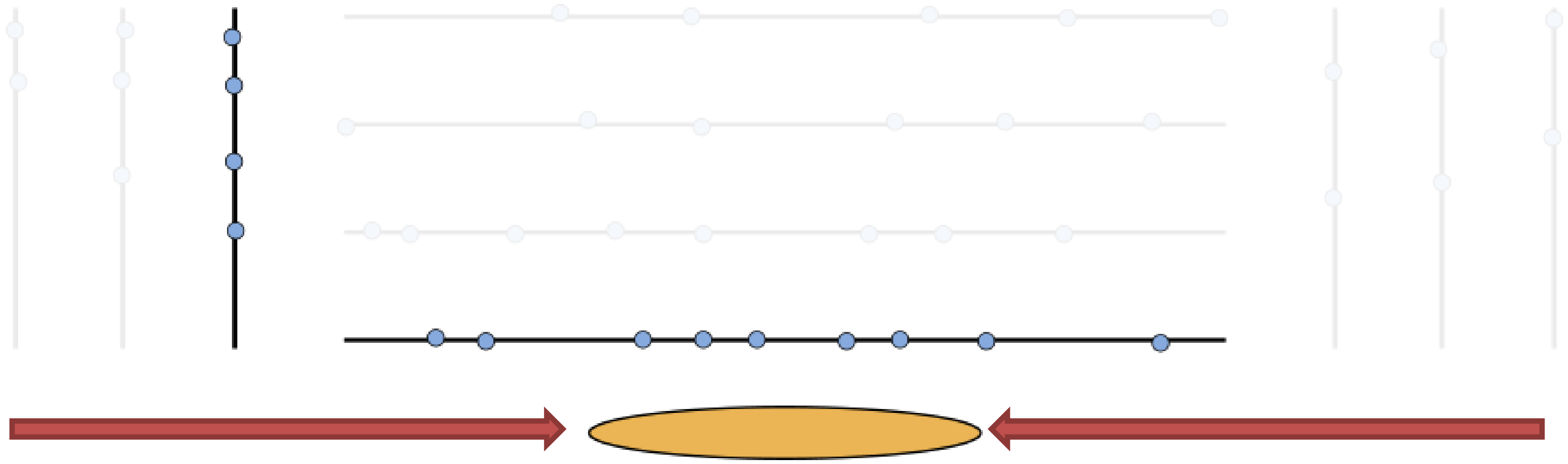
RMS HEP Algorithm

- 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



RMS HEP 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
- Repeat until happy…
- Does this fit the idea of massively parallel computation? I don't really think so…



RMS HEP Algorithm

This kind of algorithm is not very suitable for GPUs:

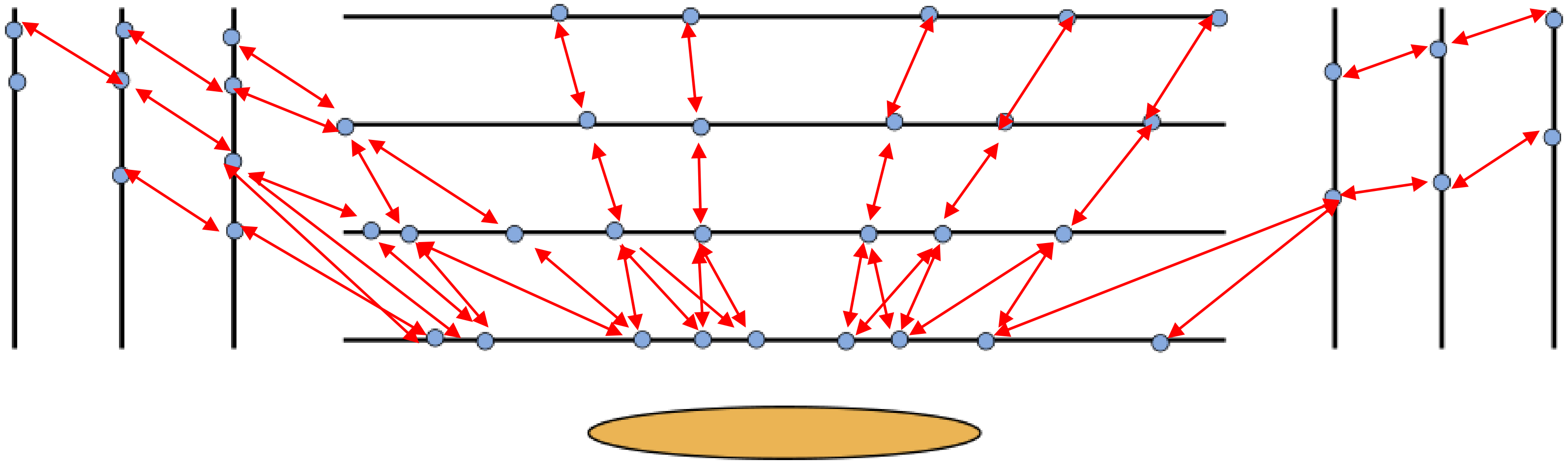
- Absence of massive parallelism
- Poor data locality
- Synchronizations due to iterative process
- Very Sparse and dynamic problem (that's the hardest part, still unsolved)
- Parallelization does not mean making a sequential algorithm run in parallel
  - It requires a deep understanding of the problem,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

The algorithm was redesigned from scratch getting inspiration from Conway's Game of Life

- Traditional Cellular Automata excluded because 2x slower
  - quadruplets by triplets sharing a doublet

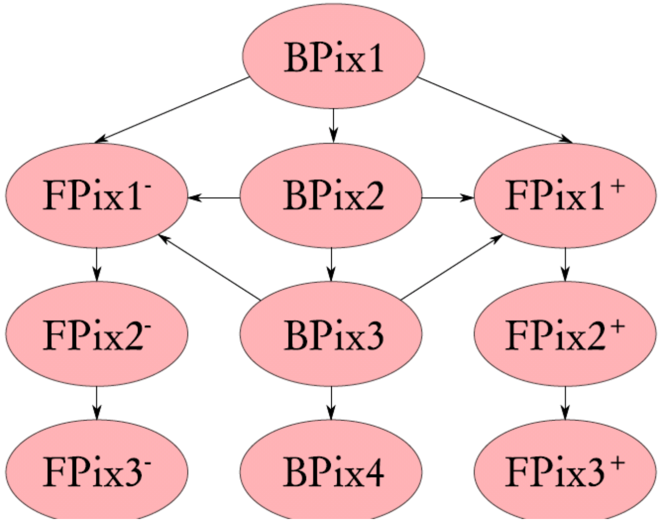
Cellular Automaton (CA)

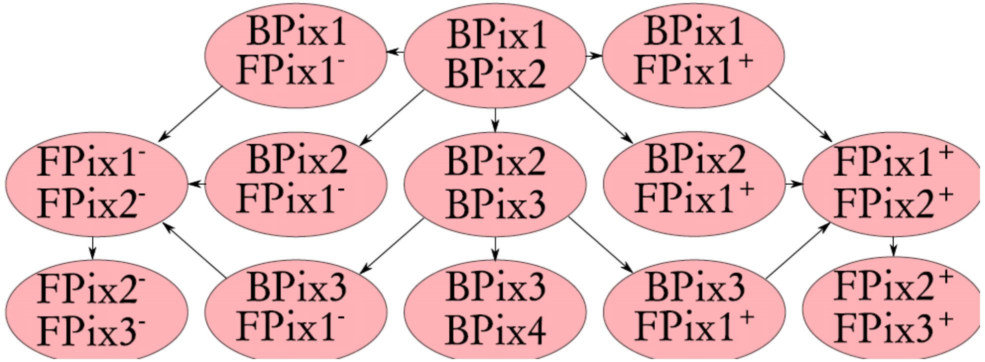
- The CA is a track seeding algorithm designed for parallel architectures
- It requires a list of layers and their pairings
  - A graph of all the possible connections between layers is created
  - Doublets aka Cells are created for each pair of layers (compatible with a region hypothesis)
  - Fast computation of the compatibility between two connected cells
  - No knowledge of the world outside adjacent neighboring cells required, making it easy to parallelize
- However this is not a static problem, not at all…



CAGraph of seeding layers

- Seeding layers interconnections
- Hit doublets for each layer pair can be computed independently by sets of threads

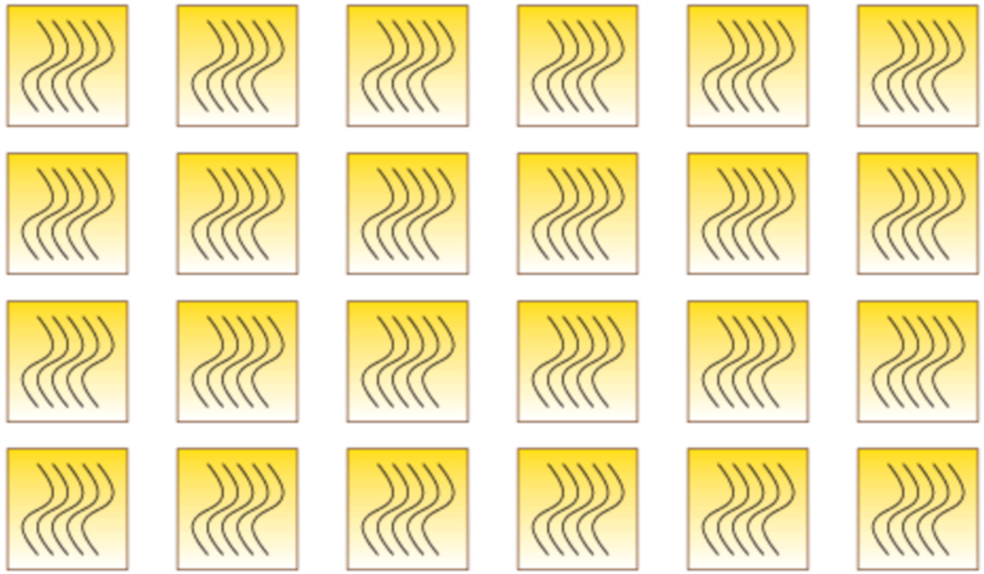




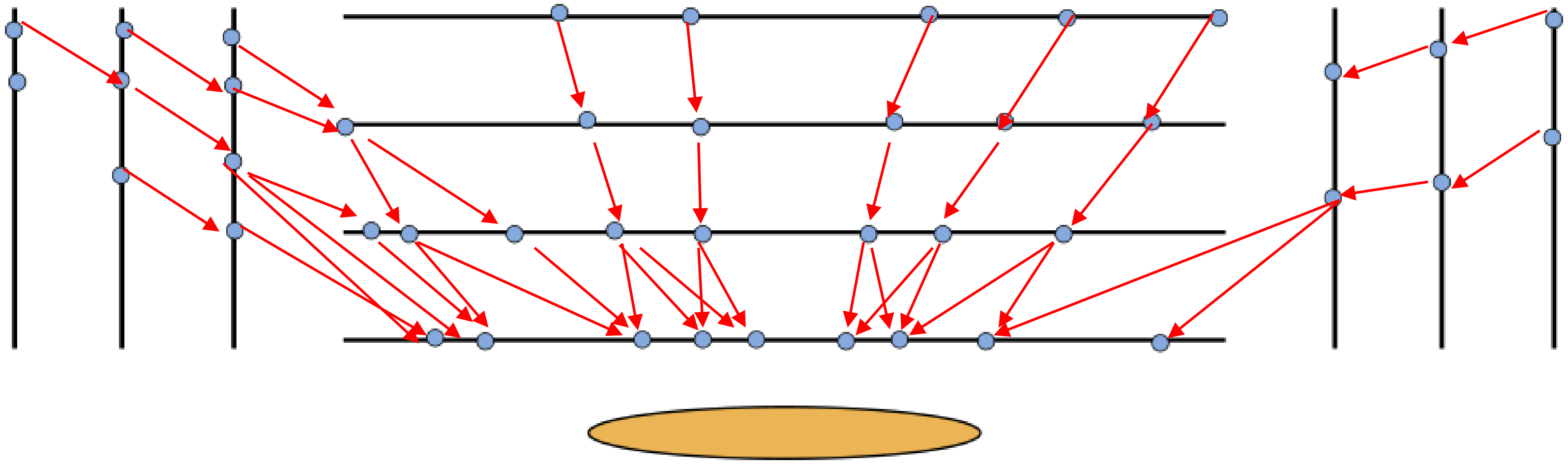
Cells Connection

blockIdx.x and threadIdx.x = Cell id in a LayerPair

blockIdx.y = LayerPairIndex [0,13)

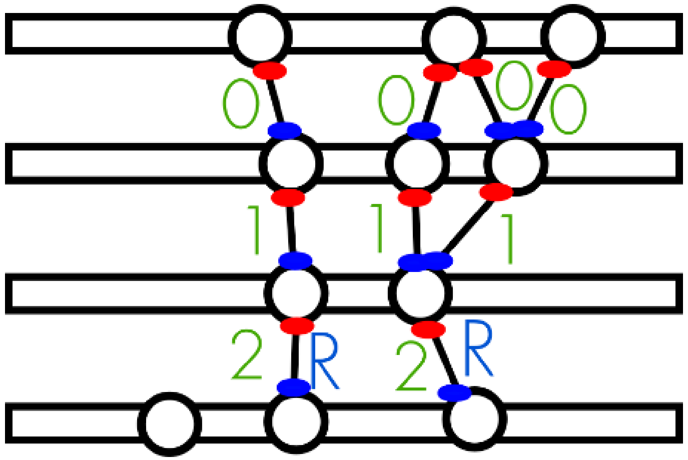


Each cell asks its innermost hits for cells to check compatibility with.

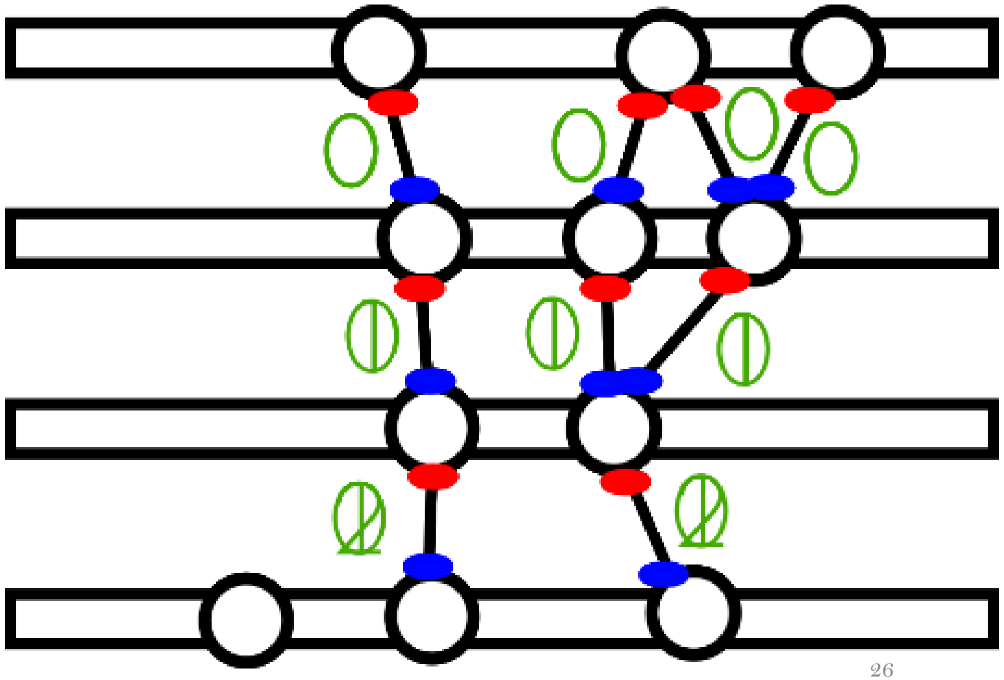


Evolution

- 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.



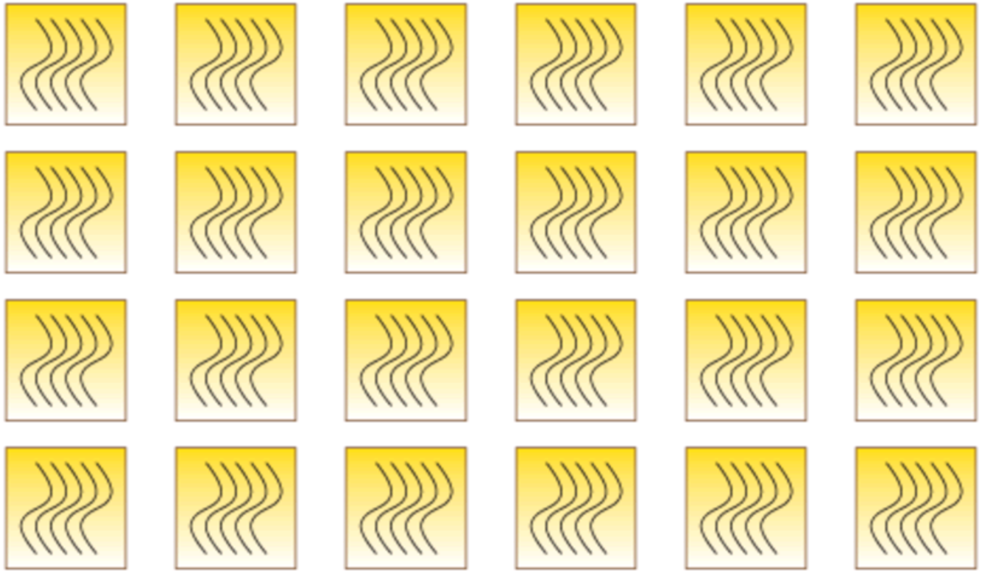
T=0



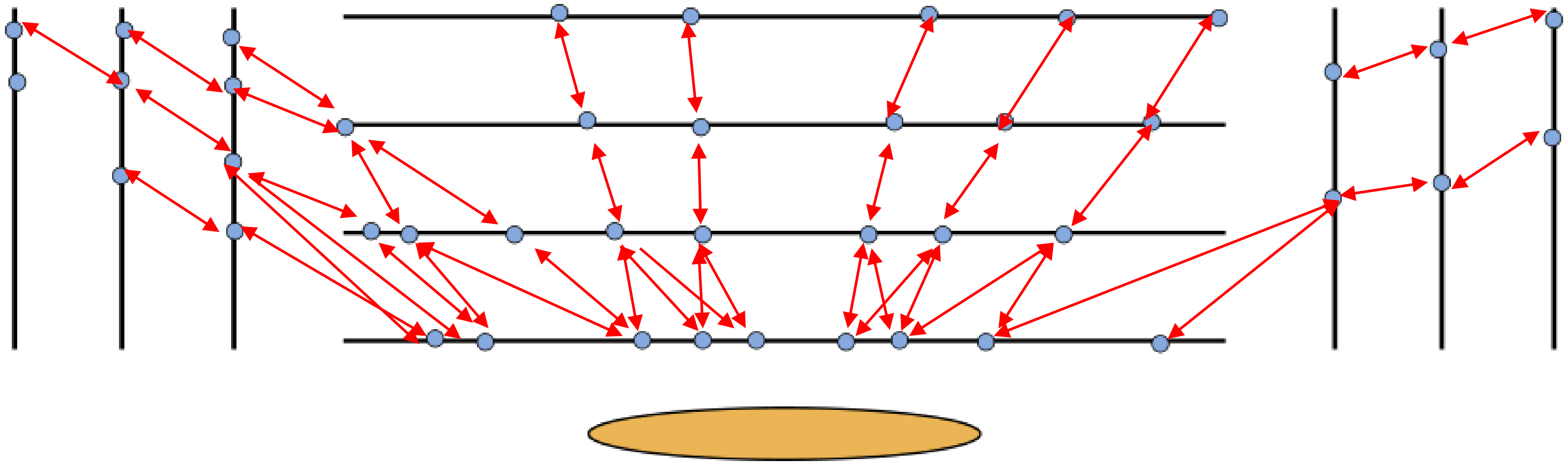
Quadruplets finding

blockIdx.x and threadIdx.x = Cell id in a Root LayerPair

blockIdx.y = LayerPairIndex in RootLayerPairs



Each cell on a root layer pair will perform a parallel DFS of depth = 4 following outer neighbors.



Integration studies

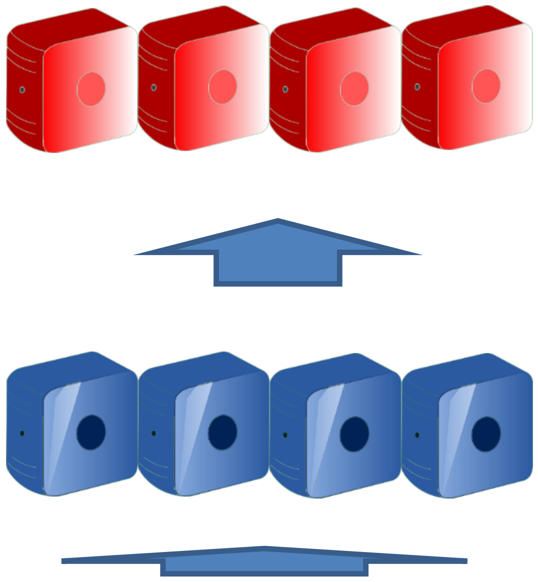
Integration in the Cloud and/or HLT Farm

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

Filter Units

Today

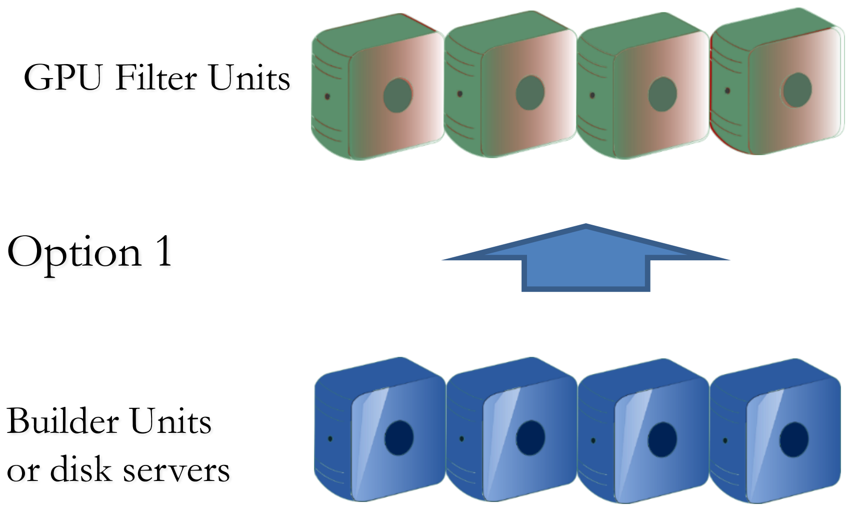
Builder Units
or disk servers



CMS FE, Read-out Units

Integration in the Cloud/Farm

- Every FU is equipped with GPUs
  - tracking for every event



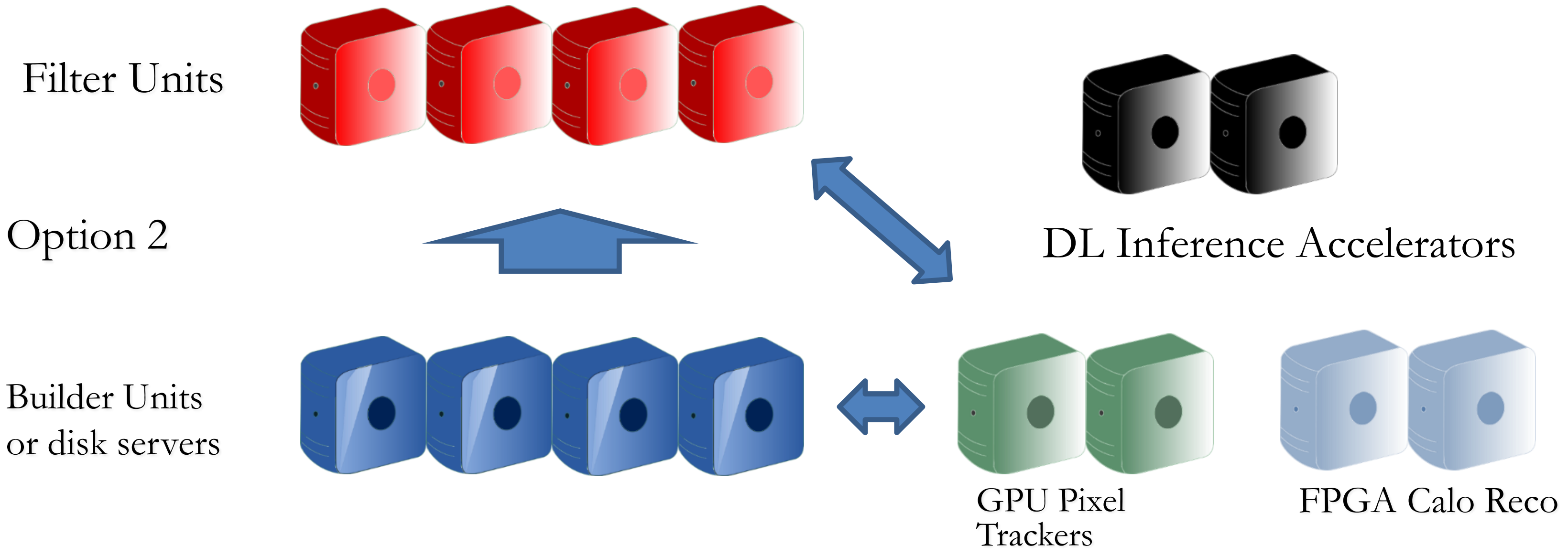
- Rigid design

  + easy to implement

  - Requires common acquisition, dimensioning etc

Integration in the Cloud/Farm

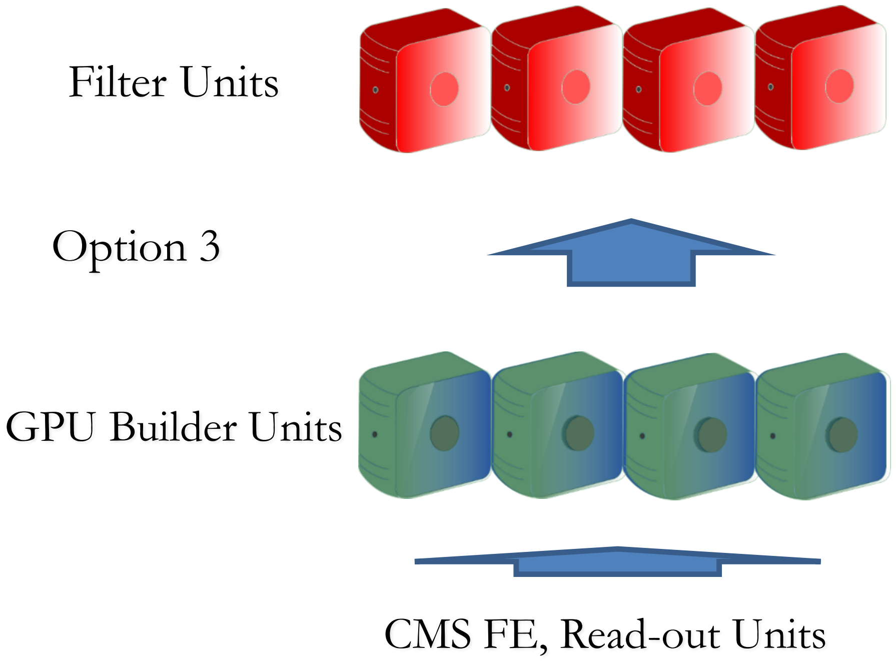
- A part of the farm is dedicated to a high density GPU cluster
- Tracks (or other physics objects like jets) are reconstructed on demand



- Flexible design
  + Expandible, easier to balance
  - Requires more communication and software development (See talk by Jean-Loup)

Integration in the HLT Farm

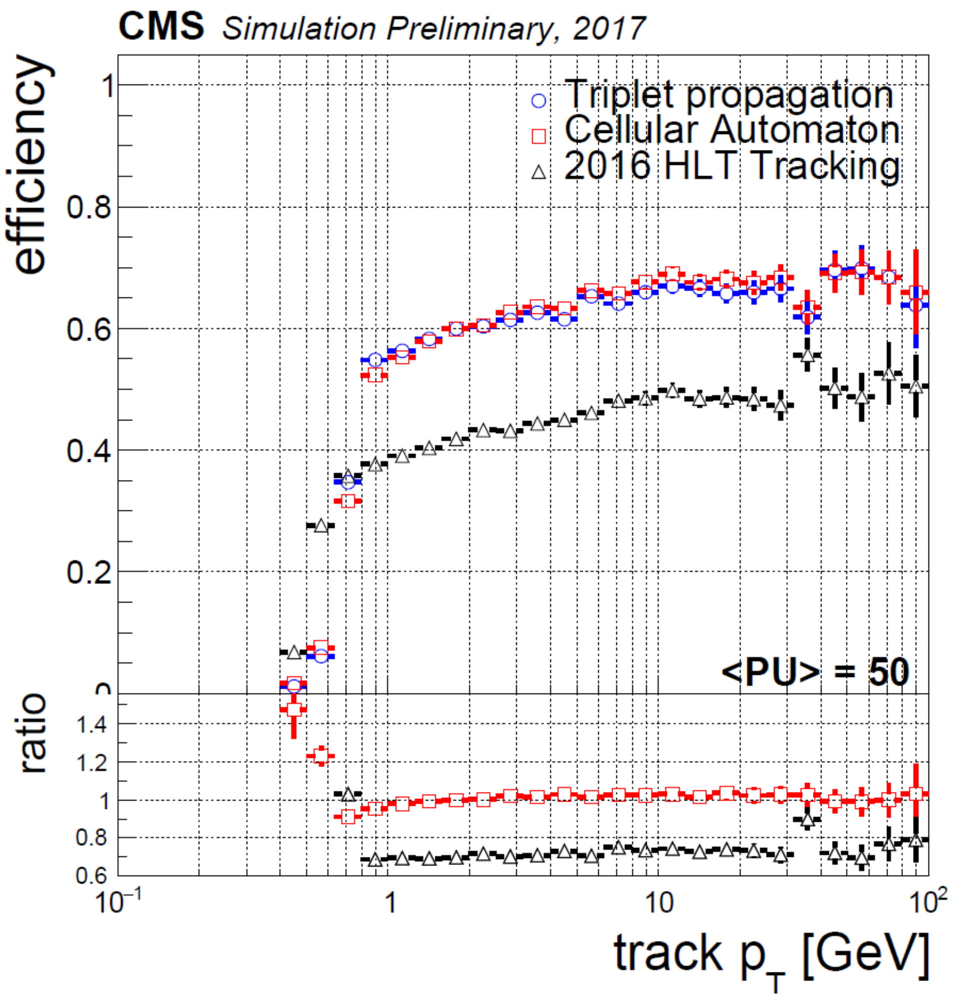
- Builder units are equipped with GPUs:
  – events with already reconstructed tracks are fed to FUs with GPUDirect
  – Use the GPU DRAM in place of ramdisks for building events.

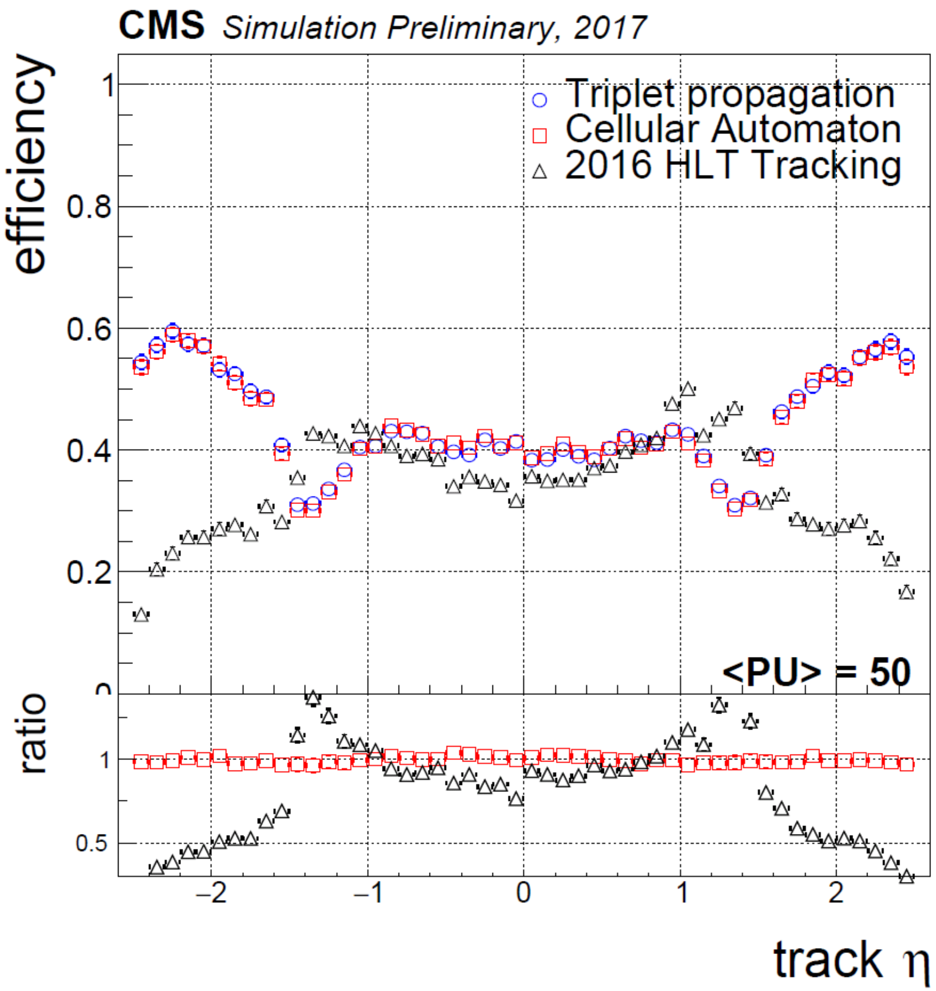


- Very specific design
  + fast, independent of FU developments, integrated in readout
  - Requires specific DAQ software development: GPU “seen” as a detector element

Tests

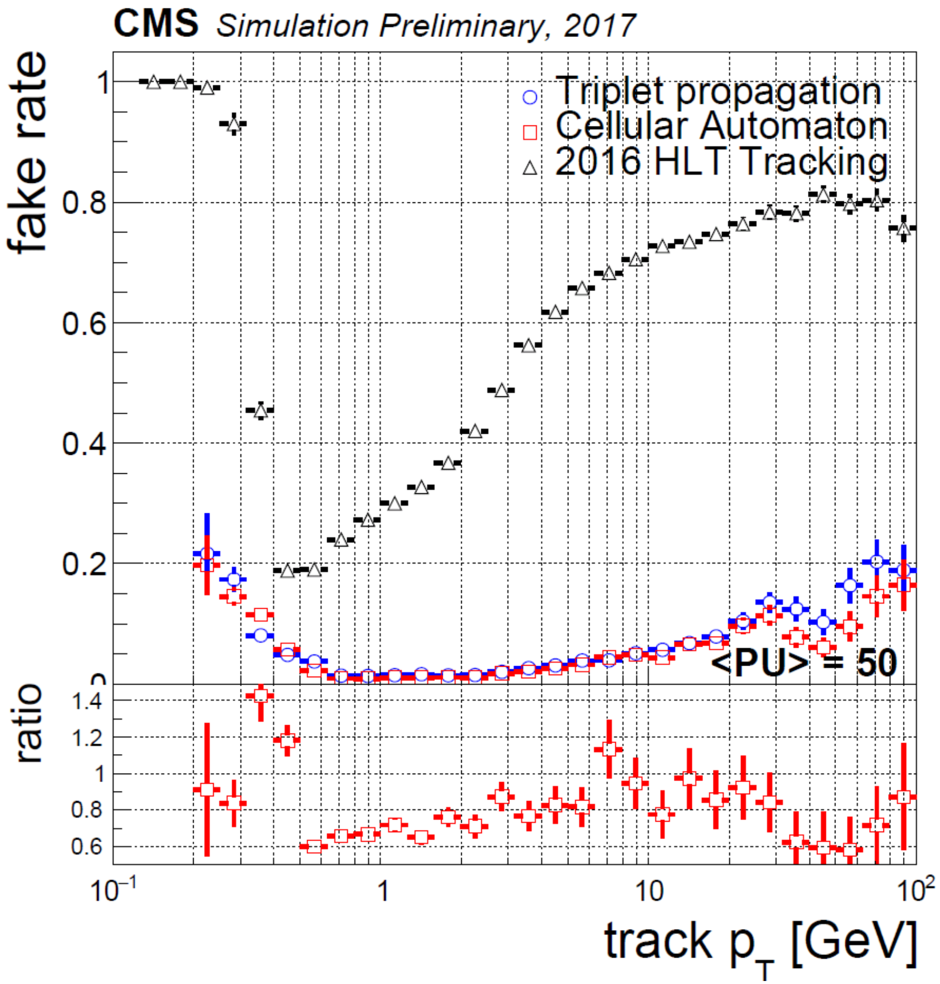
Simulated Physics Performance PixelTracks

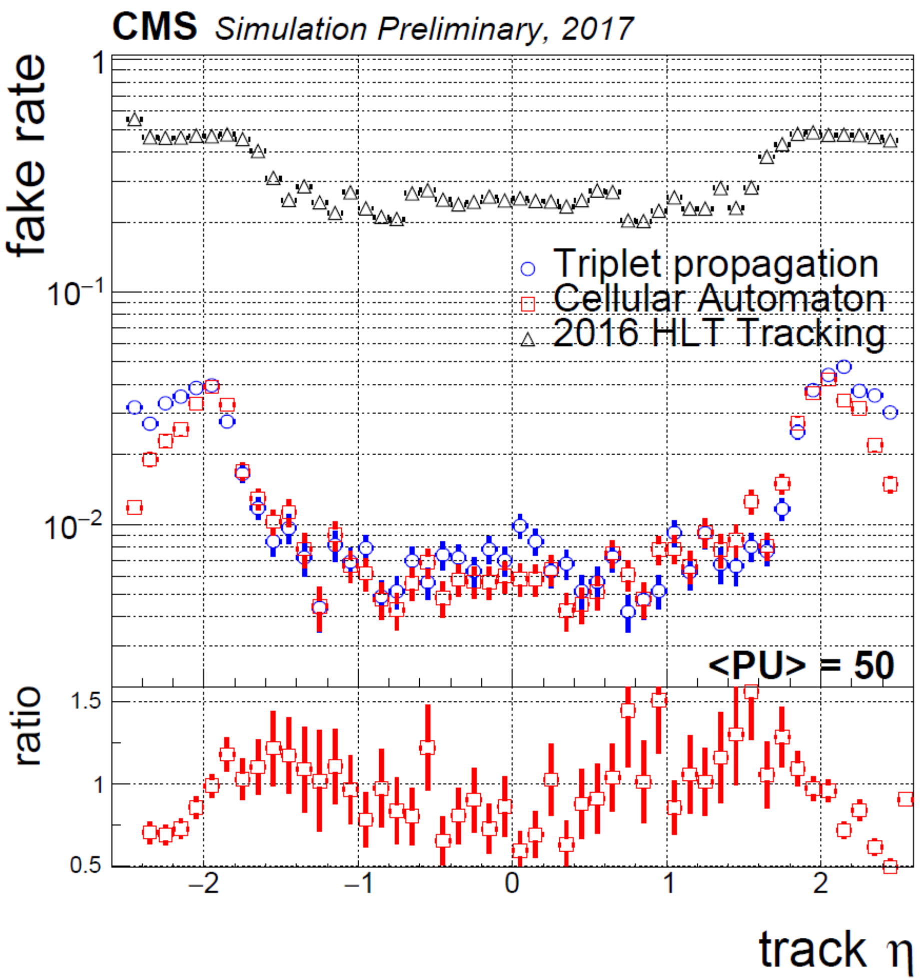




- CA tuned to have same efficiency as Triplet Propagation
- Efficiency significantly larger than 2016, especially in the forward region ($|\eta|>1.5$).

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

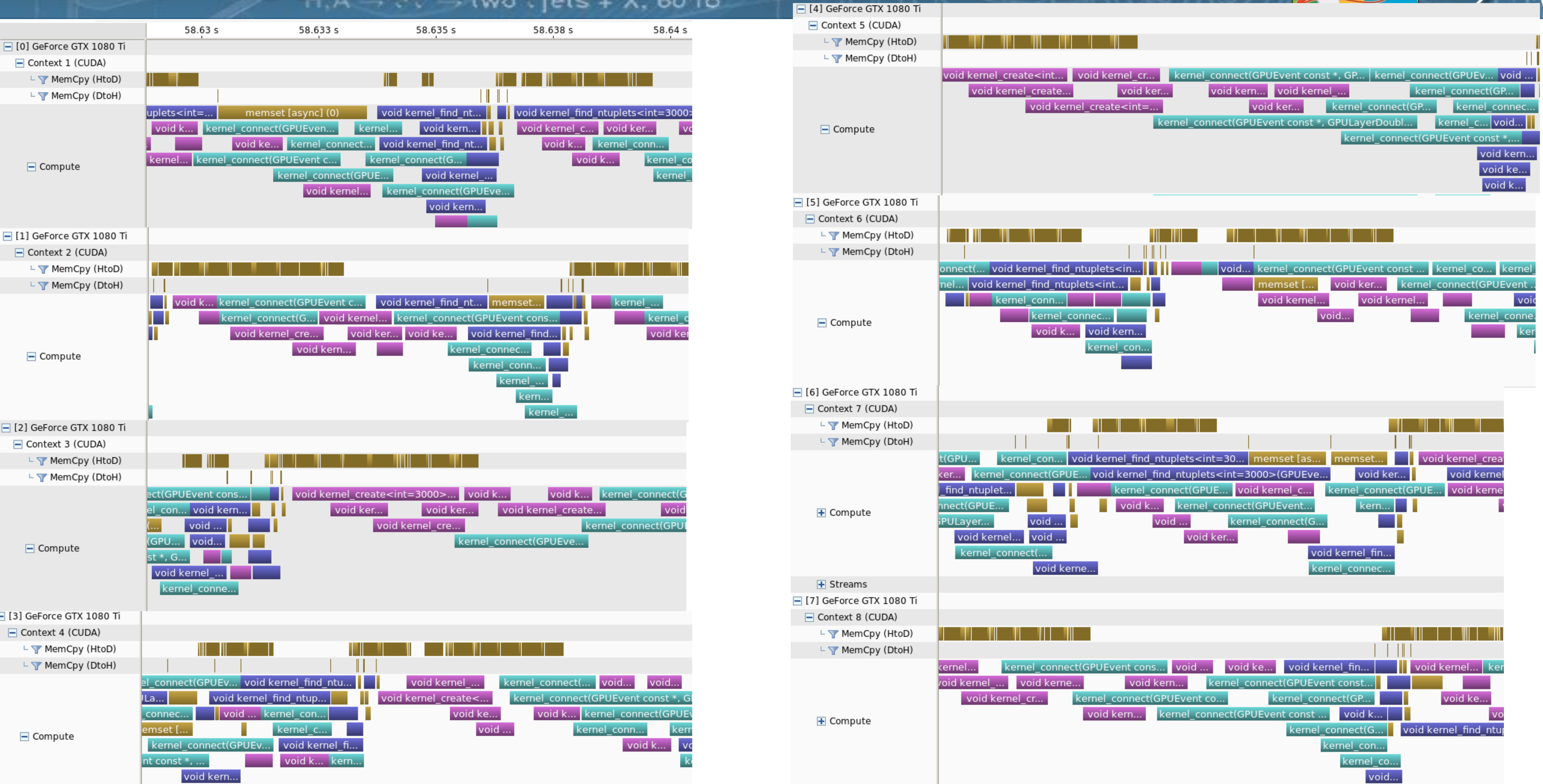
Hardware on the bench

- We acquired small machine for development and testing:
  - 2 sockets x Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz (12 physical cores)
  - 256GB system memory
  - 8x GPUs NVIDIA GTX 1080Ti
  - Total cost: 5x 🍌

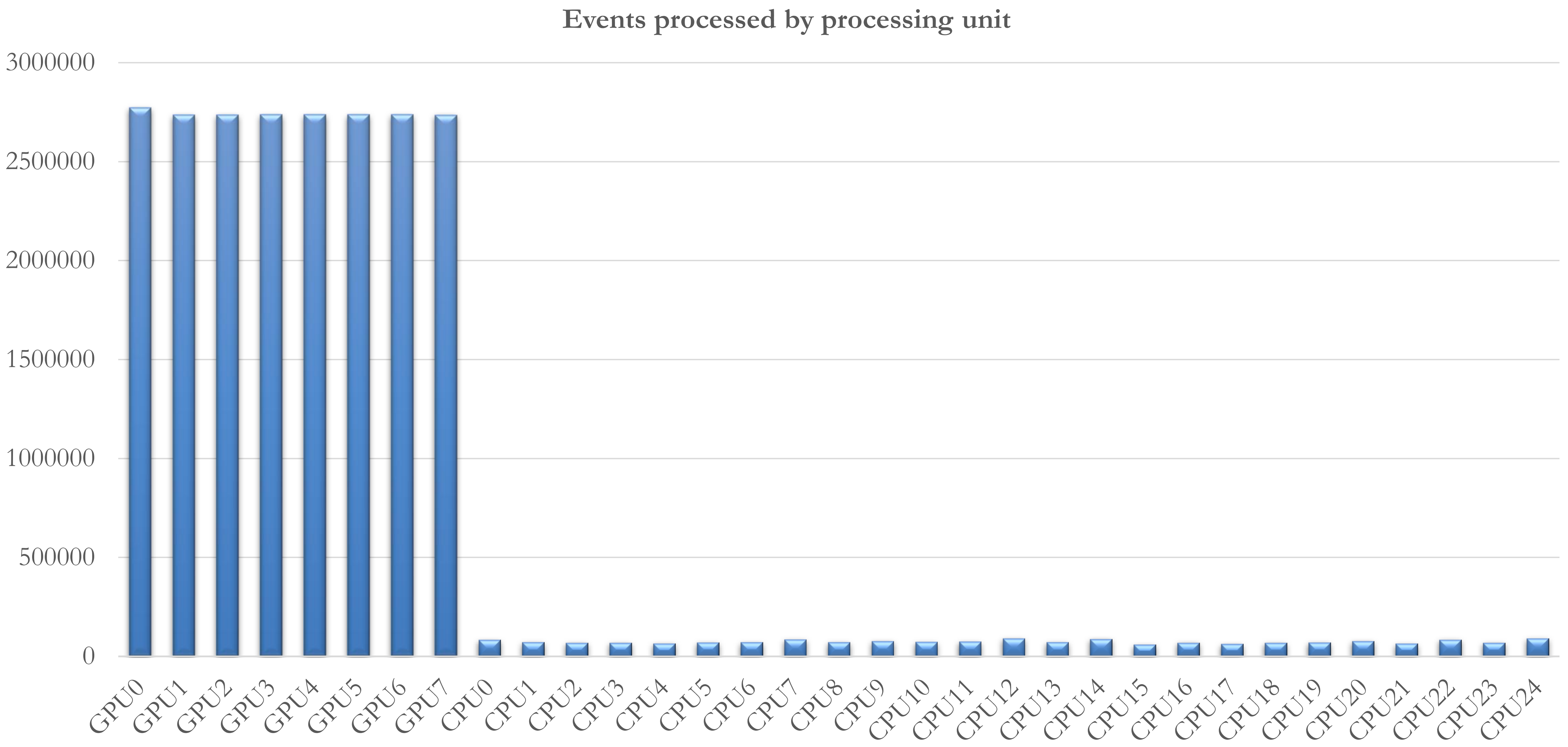
Rate test

- 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

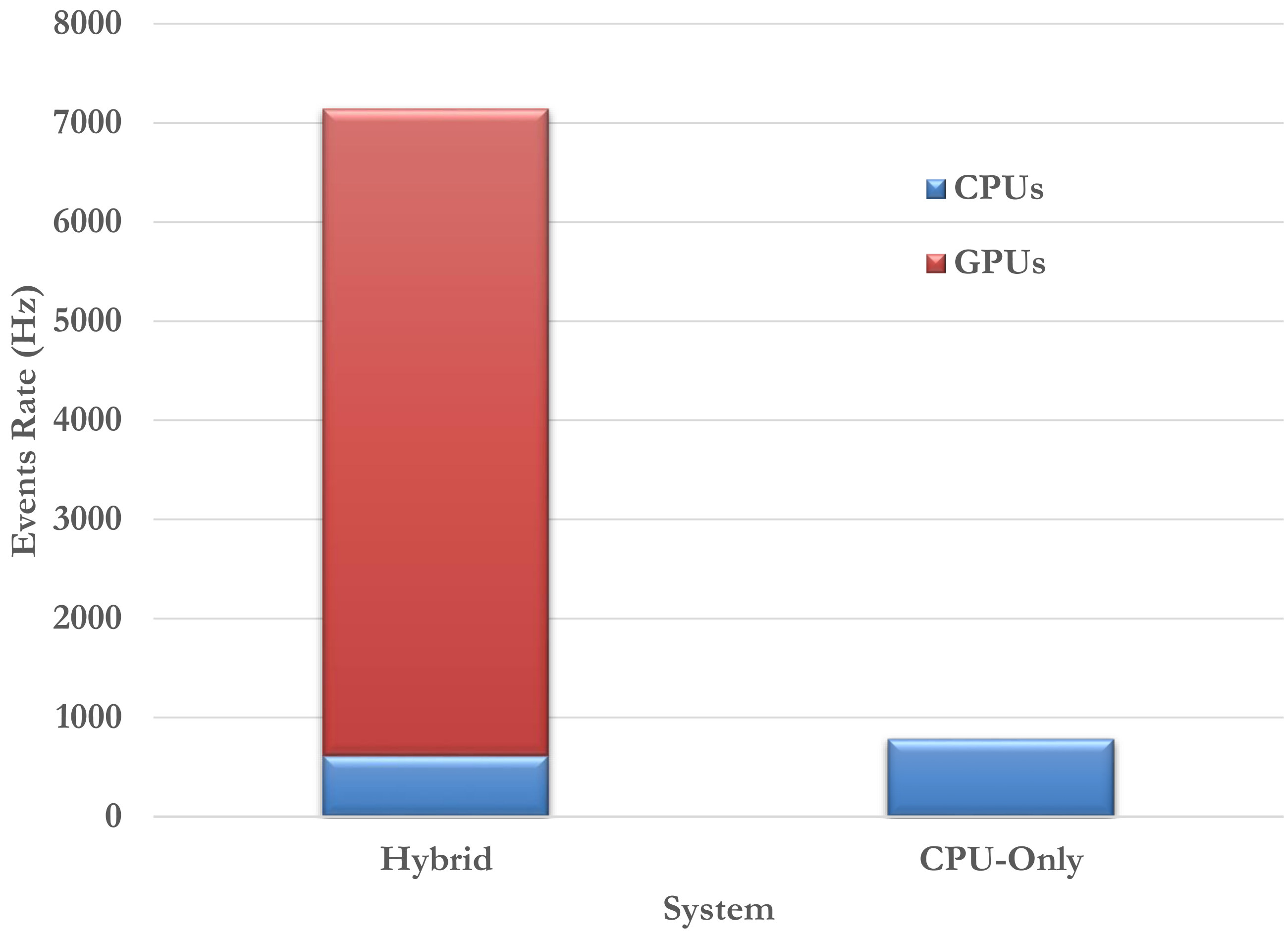


Events processed by processing unit
3000000
2500000
2000000
1500000
1000000
500000
0
GPU0
GPU1
GPU2
GPU3
GPU4
GPU5
GPU6
GPU7
CPU0
CPU1
CPU2
CPU3
CPU4
CPU5
CPU6
CPU7
CPU8
CPU9
CPU10
CPU11
CPU12
CPU13
CPU14
CPU15
CPU16
CPU17
CPU18
CPU19
CPU20
CPU21
CPU22
CPU23
CPU24

Rate test

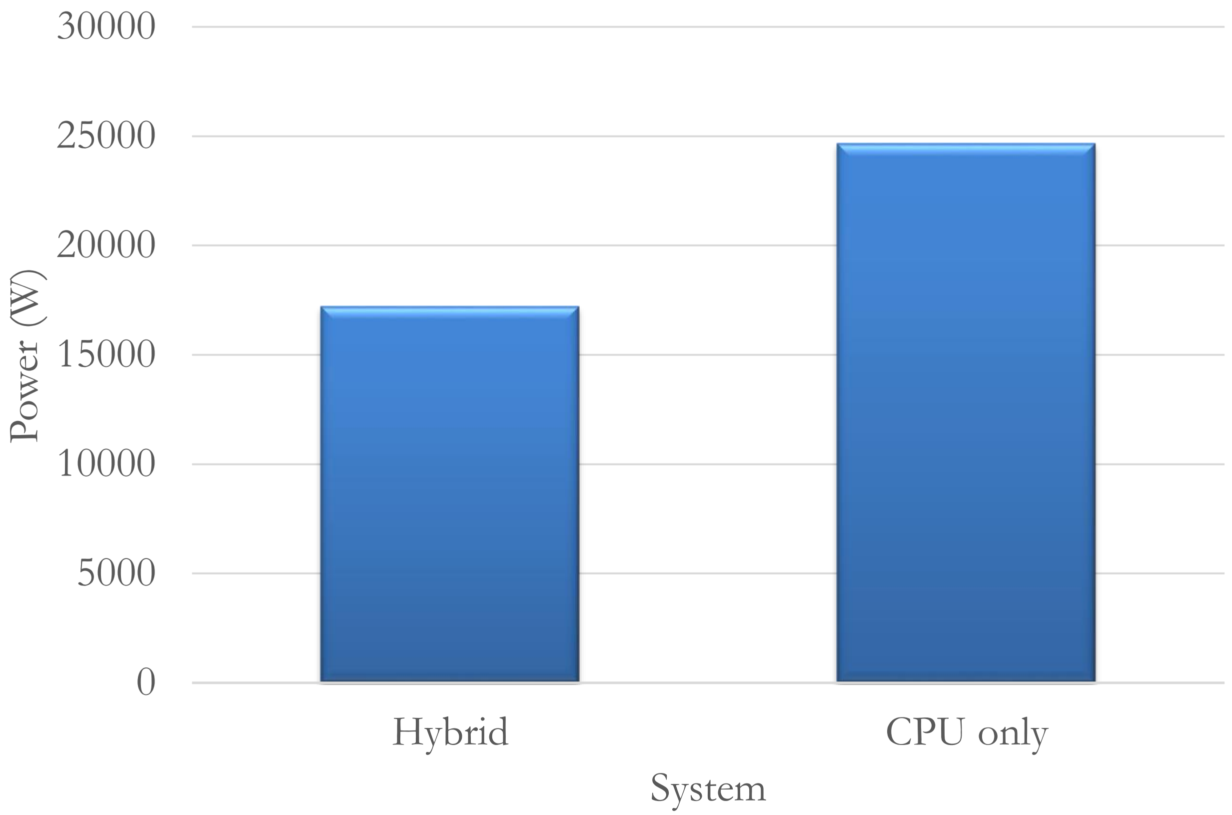
- Total rate measured:
  - 8xGPU: 6527 Hz
  - 24xCPUs: 613 Hz
- Number of nodes to reach 100kHz: ~14
- Total Price: 70x 🍌

- 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



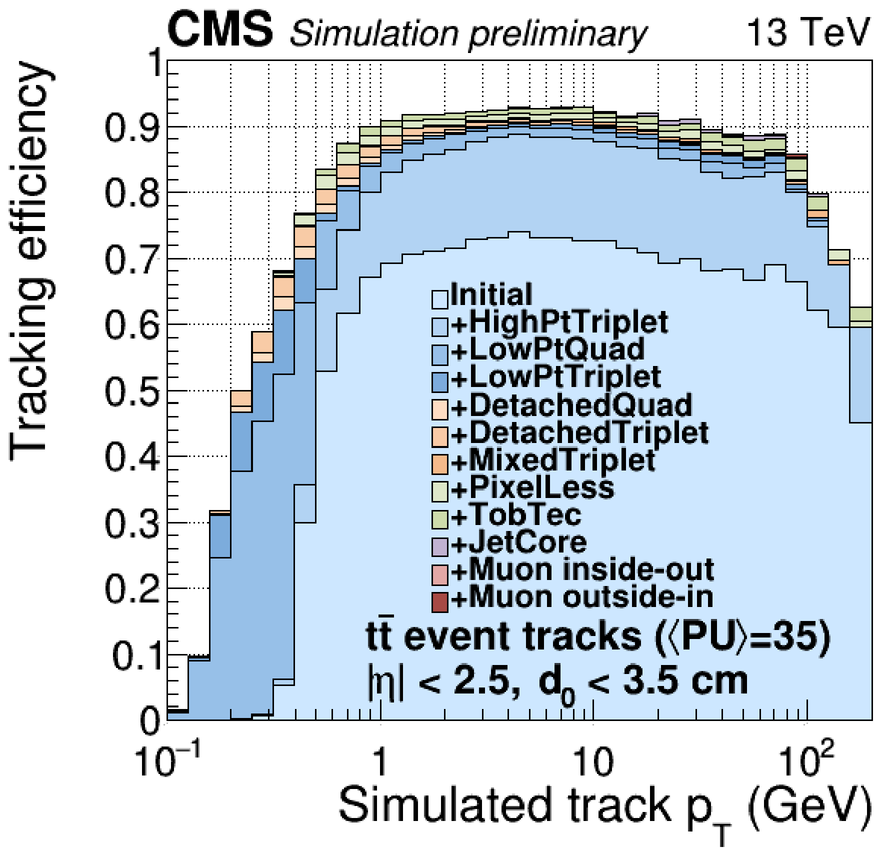
- That is 1/3 more 🍌s in the energy bill when processing 100kHz input

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

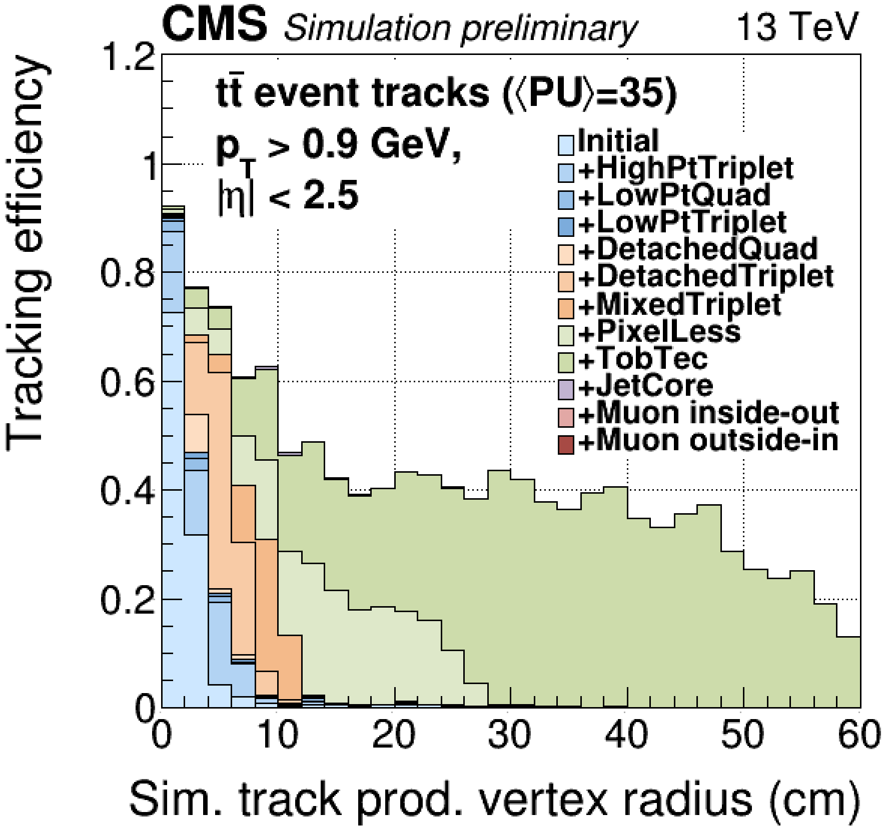
Iteration	Seeding	Target track
Initial	pixel quadruplets	prompt, high p_T
LowPtQuad	pixel quadruplets	prompt, low p_T
HighPtTriplet	pixel triplets	prompt, high p_T recovery
LowPtTriplet	pixel triplets	prompt, low p_T recovery
DetachedQuad	pixel quadruplets	displaced--
DetachedTriplet	pixel triplets	displaced-- 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



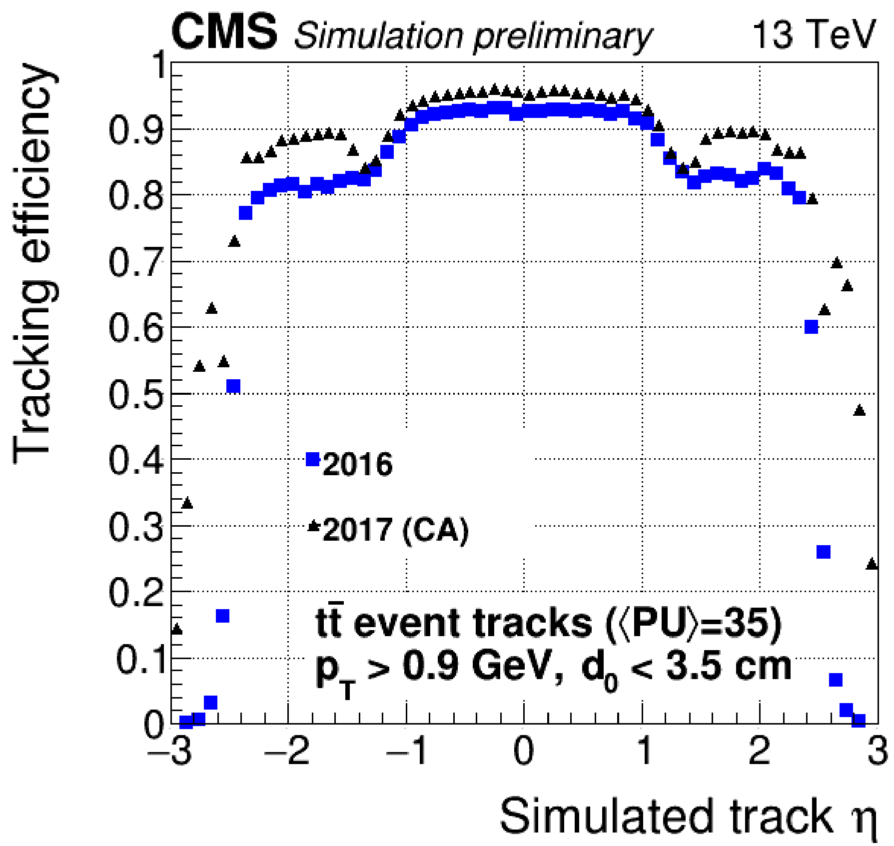
CA in offline tracking

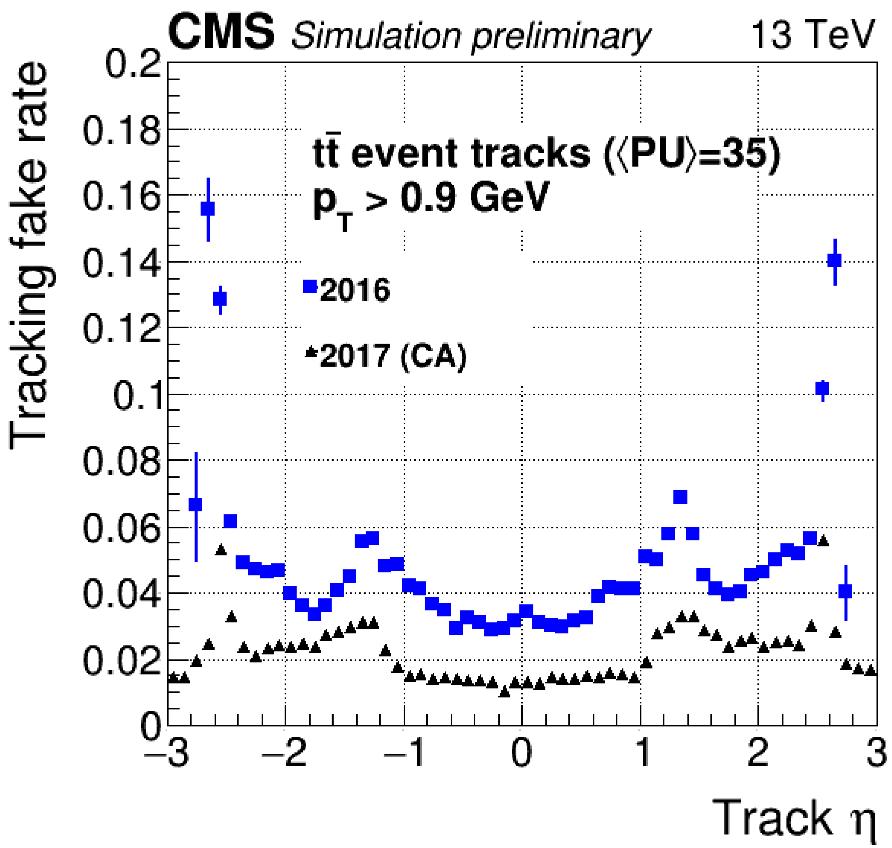
- The performance of the sequential Cellular Automaton at the HLT justified its integration also in the 2017 offline iterative tracking

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

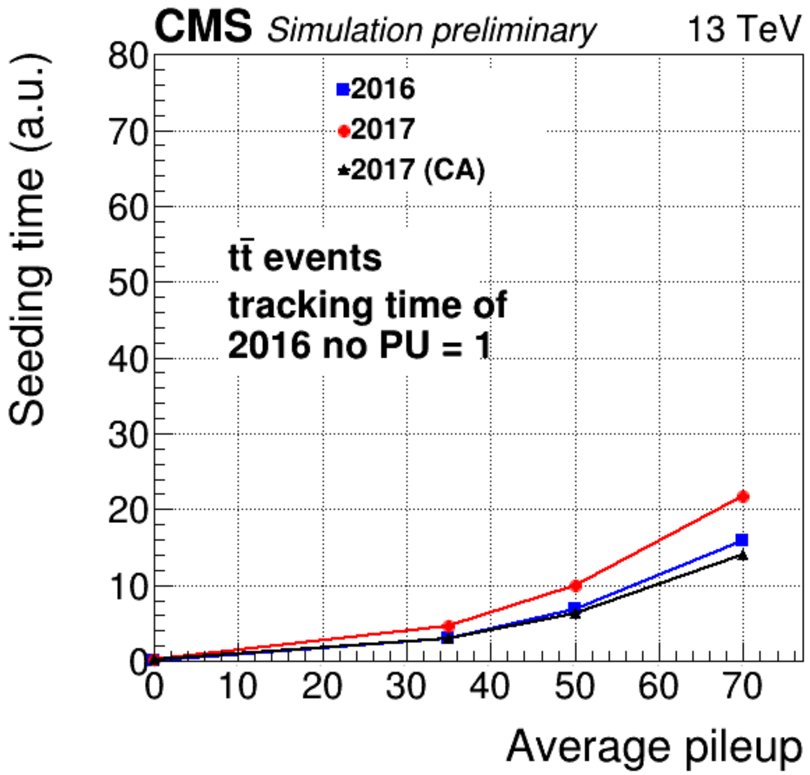




- Reconstruction efficiency increased
  - especially in forward region.
- Fake rate significantly reduced in the entire pseudo-rapidity range

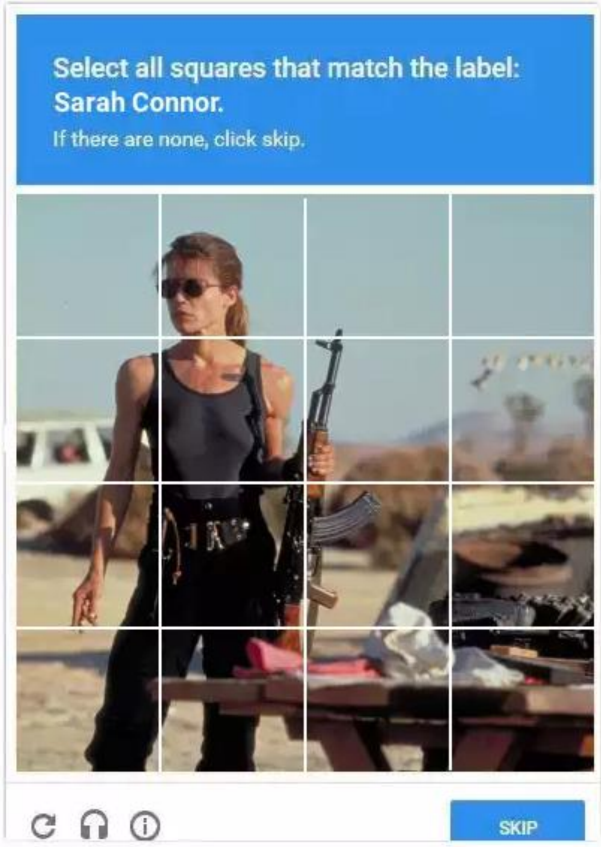
Timing vs PU

- 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
- Vincenzo replaced the CMS Phase2 offline track seeding with sequential CA
  - Overall tracking 2x faster at PU200
  - T(PU=200) = 4xT(PU50)
    - Detector and algorithms defeated combinatorial complexity



Conclusion

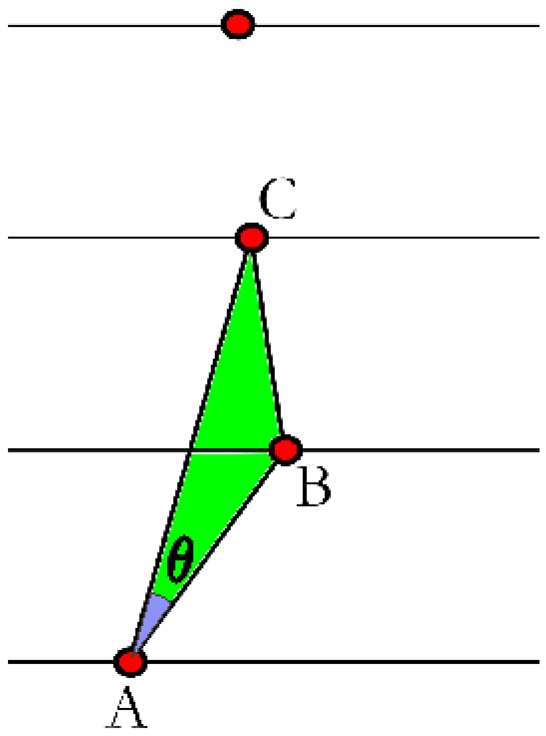
- Pixel Track seeding algorithms have been redesigned with high-throughput parallel architectures in mind
- Improvements in performance may come even when running sequentially
  - Factors at the HLT, tens of % in the offline, depending on the fraction of the code that use new algos
- A library is being created under HSF following the demand of ACTS and LHCb:
  - Trick Track V0.1
- Graph-based algorithm are very powerful
  - By adding more Graph Theory sugar, steal some work from the track building and become more flexible
- The GPU and CPU algorithms run in CMSSW and produce the same bit-by-bit result
  - Transition to GPUs@HLT during Run3 smoother
- Running Pixel Tracking at the CMS HLT for every event would become cheap @PU ~ 50 – 70
  - Integration in the CMS High-Level Trigger farm under study
- DNNs under development for early-rejection of doublets based on their cluster shape and track classification



Back up

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'