Jet trigger algorithms with GPUs for the ATLAS Upgrade

Ademar Delgado[1,2], José Silva[1,3]

A. Gomes[1,2], P. Conde[1,2], J. Augusto[1,2], A. Pina[1,3], J. Rufino[1,3]

[1]LIP, Laboratório de Instrumentação e Física Experimental de Partículas
[2]FCUL, Faculdade de Ciências da Universidade de Lisboa
[3]ECUM, Escola de Ciências da Universidade do Minho

19 de Fevereiro de 2016

The Jet trigger challenges at the LHC:

Jet:

- Collimated spray of particles initiated by quarks or gluons
- Important for QCD studies, SM physics and search for new physics (SUSY,...)

Trigger system:

- 40 MHz event rate
- Volume of data/s ~PB/s
- Trigger systems designed to select interesting physics events, for storage and offline analysis

	Run 2	LS2 - Phase-I	Run 3
	2015-18		2020-22
Centre of mass Energy $\sqrt{s}$ (TeV)	13-14		14
Luminosity ($\mathrm{cm}^{-2}\mathrm{s}^{-1}$)	1×10^{34}		2×10^{34}
Bunch spacing (ns)	25		25
Number of interaction/event	~27		~55 - 80
Total Integrated luminosity (fb^{-1})	~ 100		~ 300

ATLAS Jet trigger for Run 3:

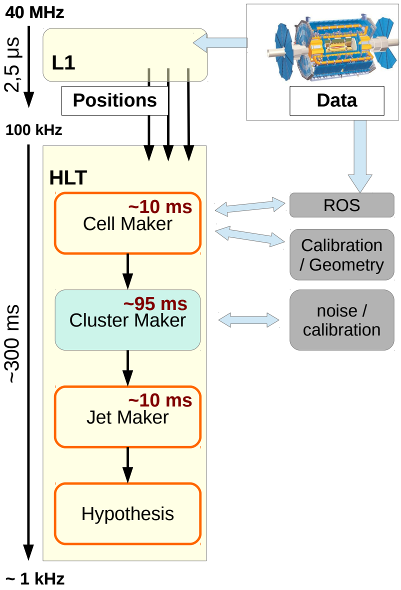
Jet energy deposits are aggregated by the reconstruction algorithms and calibrated to provide the jet momentum measurement.

In Run 3:

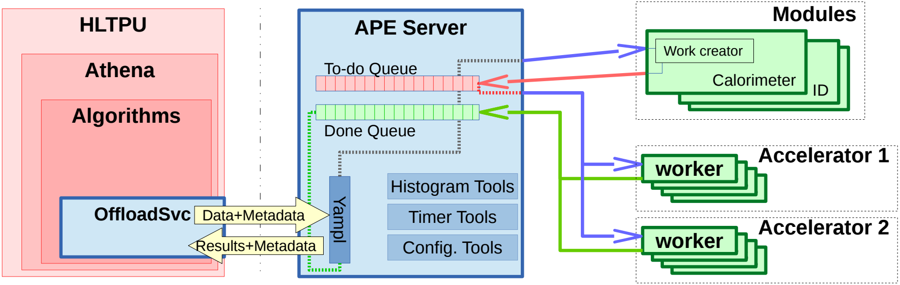
- Up to 80 collisions per event
- Advanced algorithms needed to better suppress pile-up
- More computation required

Evaluate the use of GPU's on the HLT

- Demonstrator prototype under implementation
  - LIP's responsibility: calorimeter trigger chains
- Needs re-design of the algorithms: single instruction - multiple data paradigm



Demonstrator architecture:



Client side

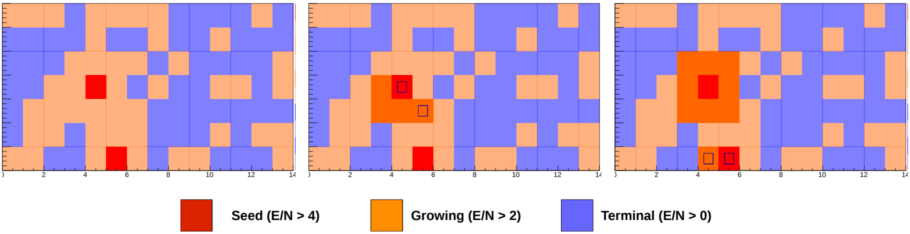
- One HLT processing unit per core
- Athena framework
    - Executes chains of algorithms
    - Provides data and monitoring services
    - Caches responses

Server side

- Independent from Athena
- Efficient accelerator resource management
- Can exploit several technologies

- Specific modules and services implemented for the calorimeter data/tasks offloading and processing

Topological Clustering algorithm:

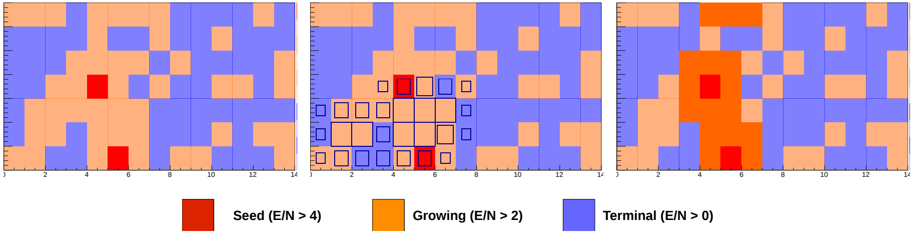


Group cells into clusters according to their Energy/Noise level.
Cluster Growing:

- Each Seed initiates a clusters
- For each Seed (and later Growing) cell:
  - Find all neighbours (3D) and add them to clusters:
    - If a neighbour is a Seed or Growing in another cluster → merge them
    - If a neighbour is a Growing cells → add it to next iteration list
- Terminal cells do not propagate the cluster

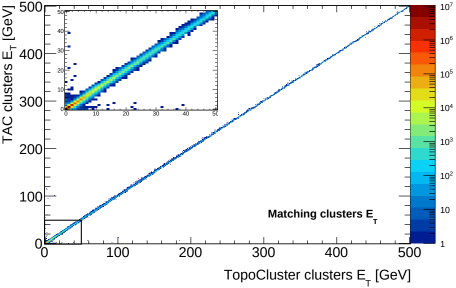
Topological Automaton Clustering algorithm:

- Designed to exploit the parallelism of the GPU



- Based on a Cellular Automaton
  - Propagate of flags on a grid of elements
- Assign unique flag (energy ordered) to each Seed cell
- Process all cells in each iteration (untill no flag changes)
  - Compare flag to all neighbours
  - Get the largest flag
- Implementation:
  - Maximize parallelism: data organized in cell-neighbour pairs
    - Only pairs with at least a Seed or Growing cell
  - Each GPU thread process a pair

Results and next steps



First results very promising:

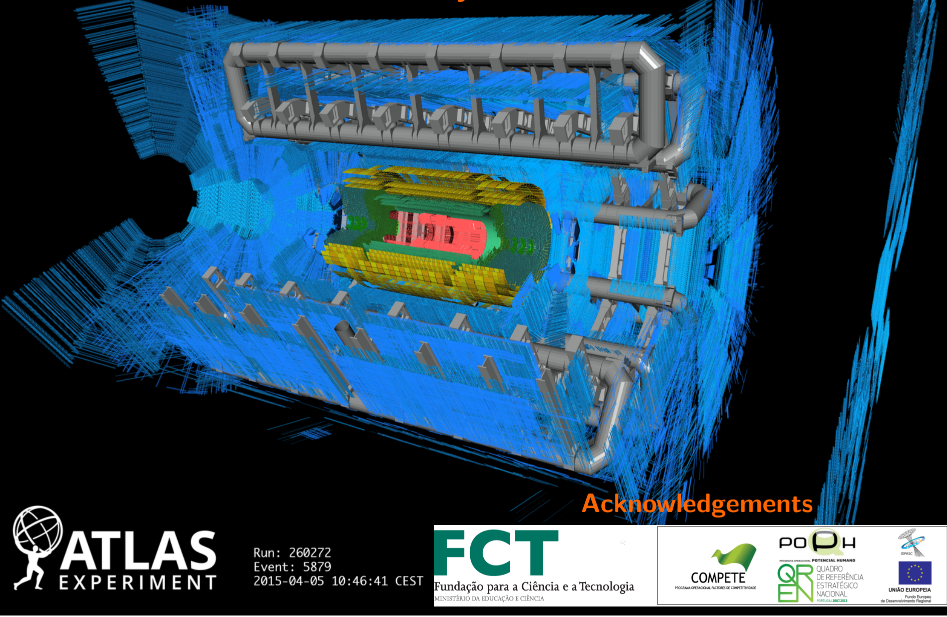
- Clusters energy correctly reconstructed
- ∼1% of TERMINAL cells associated to different clusters
    - Expected due to small differences on GPU algorithm

- Processing time (di-jet events):
    - Cluster Growing CPU ∼ 22 ms → GPU ∼ 8 ms (∼2.75X faster)
    - Data conversion overhead ∼ 12 ms
    - Optimization ongoing

Next steps

- Cluster Splitting parallelisation just started (Braga)
- Finish optimization and evaluation of the demonstrator prototype
    - Next 2-3 months
- ATLAS will take a decision on whether to use GPU's for Phase I HLT Upgrade based on these results

Thank you!



Acknowledgements

FCT Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA EDUCAÇÃO E CIÊNCIA

COMPETE
PROGRAMA OPERACIONAL FACTORES DE COMPETITIVIDADE

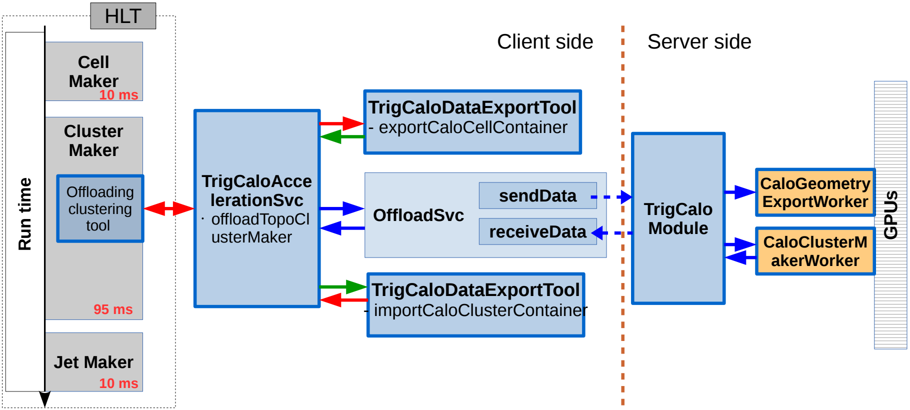
POPH
PROGRAMA OPERACIONAL POTENCIAL HUMANO

QREN
QUADRO DE REFERÊNCIA ESTRATÉGICO NACIONAL
PORTUGAL 2007.2013

IDPASC

UNIÃO EUROPEIA
Fundo Europeu de Desenvolvimento Regional

Calorimeter demonstrator:



⟷ Standard Athena EDM

⟷ lightweight EDM: structure from Acceleration EDM

⟷ APE buffer container: “wrapper” around buffer with lightweight EDM data

🟦 CPU

🟧 GPU

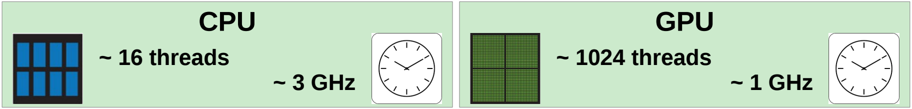
Trigger GPU demonstrator for the Phase I Upgrade:

Trigger farm hardware rolling replacement during shut-down

- New architectures need to be evaluated, parallel computing paradigm growing
- General-purpose computing on Graphics Processing Units (GPUs)

GPUs are massive parallel processing architectures

- single instruction multiple data



The Trigger GPU demonstrator comprises prototype of HLT algorithms

- Inner Detector tracking: (RAL)
- Calorimeter clustering: (LIP)
- Muon track finding: (Bologna, Rome)

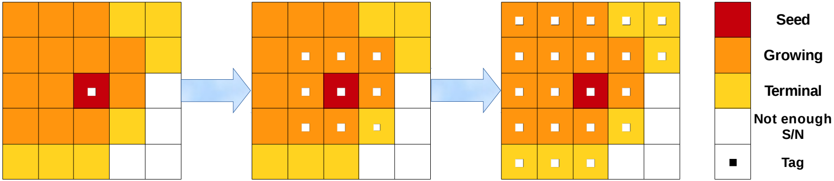
The goal is to assess the potential of GPUs for use in the HLT farm

- Main metric to be the throughput per unit cost

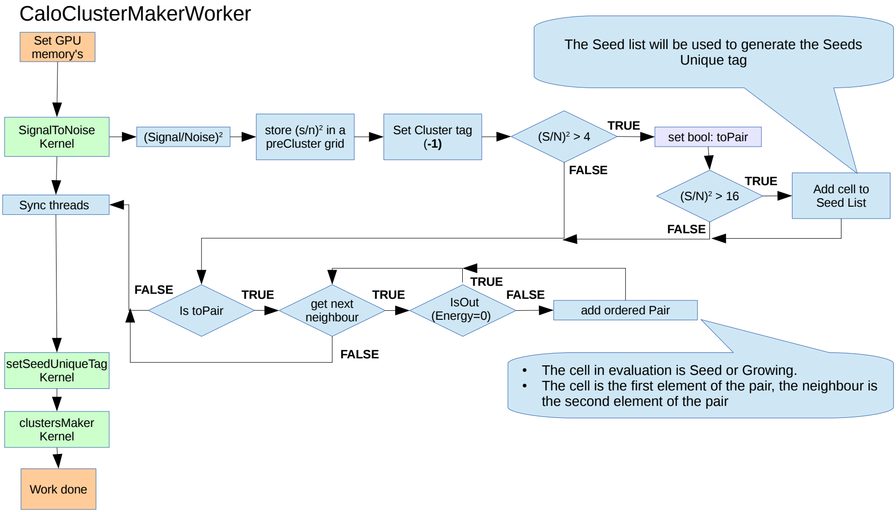
Topological Automaton Clustering:

Based on a Cellular Automaton algorithm

1. Cell signal/noise rate
2. Fill a vector of cell pairs. (reduces branching due to irregular # neighbour)
   - Only Seed or Growing cells generate pairs. (data reduction)
3. Orders the Seeds (S/N), cluster tag is the Seed position in the list
4. Loop all cell pairs proposing the highest tag as new tag till all tags remain unchanged:
   - If no/same tag proposed: continues.
   - If has one proposed tag from a Seed/Growing cell: accept tag.
   - If both cells propose a valid tag: accept the highest S/N tag. (merge clusters)



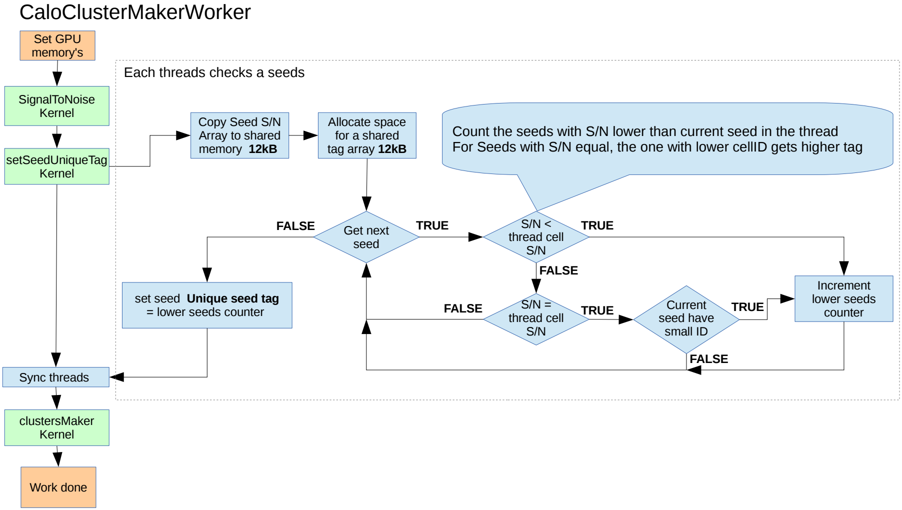
Topological Cluster alternatives: GPU CellRating



The Seed list will be used to generate the Seeds Unique tag

- The cell in evaluation is Seed or Growing.
- The cell is the first element of the pair, the neighbour is the second element of the pair

Topological Cluster alternatives: GPU Set seed unique tag


CaloClusterMakerWorker

Set GPU memory's

SignalToNoise Kernel

setSeedUniqueTag Kernel

Each threads checks a seeds

Copy Seed S/N Array to shared memory **12kB**

Allocate space for a shared tag array **12kB**

Count the seeds with S/N lower than current seed in the thread
For Seeds with S/N equal, the one with lower cellID gets higher tag

Get next seed

FALSE

TRUE

set seed **Unique seed tag** = lower seeds counter

S/N < thread cell S/N

TRUE

FALSE

S/N = thread cell S/N

FALSE

TRUE

Current seed have small ID

TRUE

FALSE

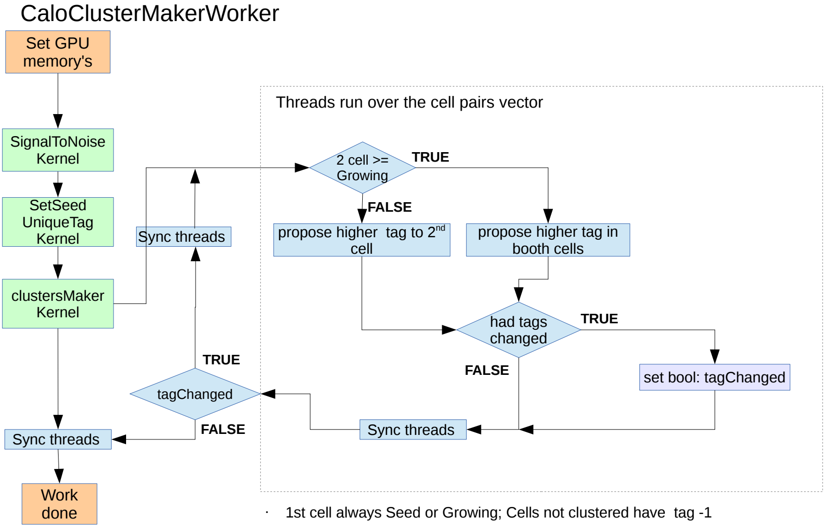
Increment lower seeds counter

Sync threads

clustersMaker Kernel

Work done

Flux diagram of TAC:



CaloClusterMakerWorker

Set GPU memory's

SignalToNoise Kernel

SetSeed UniqueTag Kernel

clustersMaker Kernel

Sync threads

Work done

Threads run over the cell pairs vector

2 cell >= Growing

TRUE

FALSE

propose higher tag to 2nd cell

propose higher tag in booth cells

had tags changed

TRUE

FALSE

set bool: tagChanged

Sync threads

tagChanged

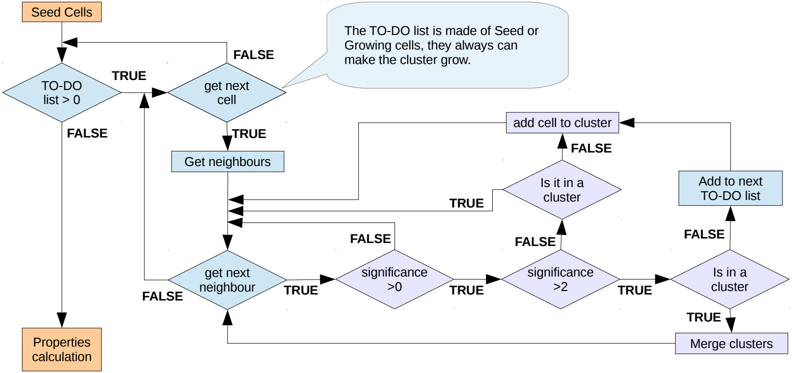
TRUE

FALSE

Sync threads

- 1st cell always Seed or Growing; Cells not clustered have tag -1

Topological Cluster alternatives: CPU Clustering



The TO-DO list is made of Seed or Growing cells, they always can make the cluster grow.