Accelerating the ATLAS Trigger System with Graphical Processing Units

7th IDPASC
Student Workshop
06/07/2022

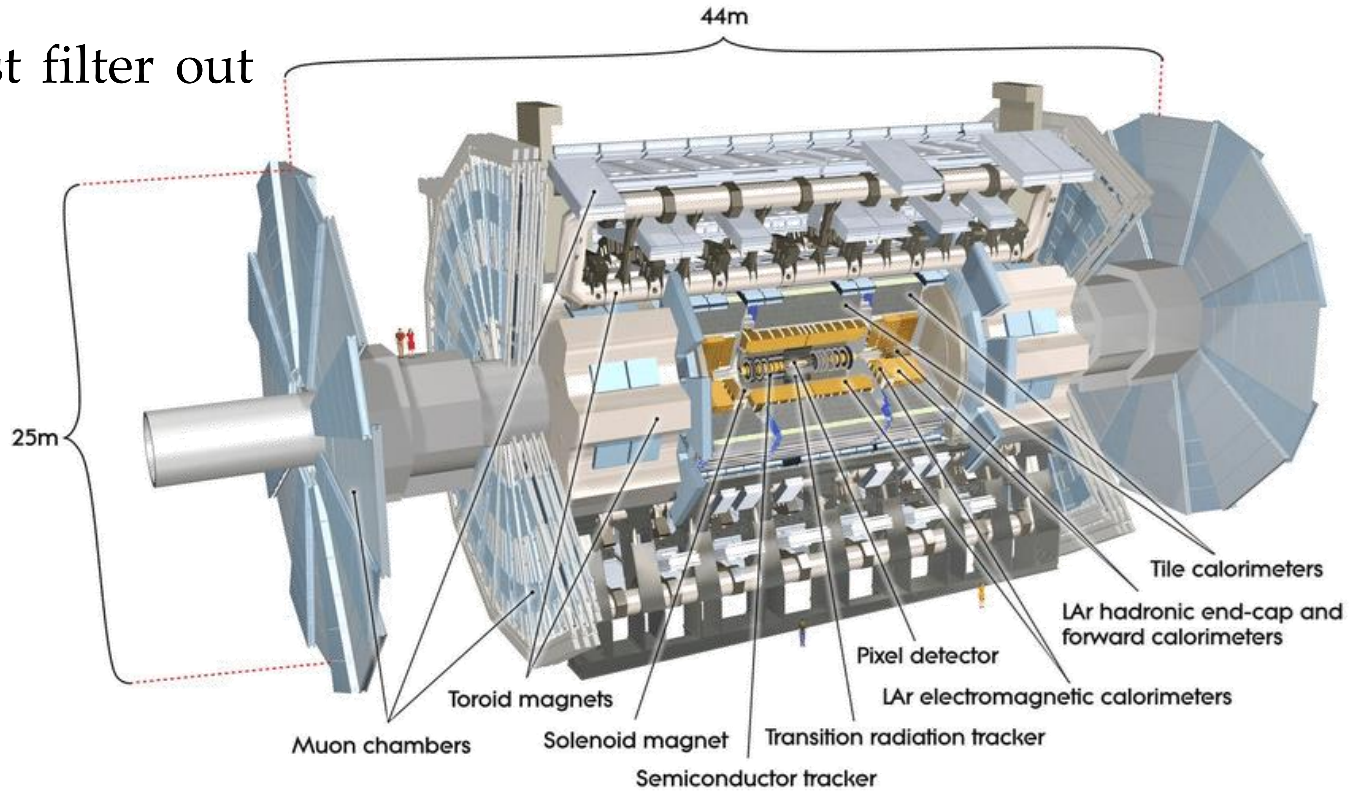
Nuno dos Santos Fernandes

REPÚBLICA PORTUGUESA
FCT Fundação para a Ciência e a Tecnologia
TÉCNICO LISBOA
LIP | LABORATÓRIO DE INSTRUMENTAÇÃO E FÍSICA EXPERIMENTAL DE PARTÍCULAS
ATLAS EXPERIMENT

ATLAS

The ATLAS Experiment

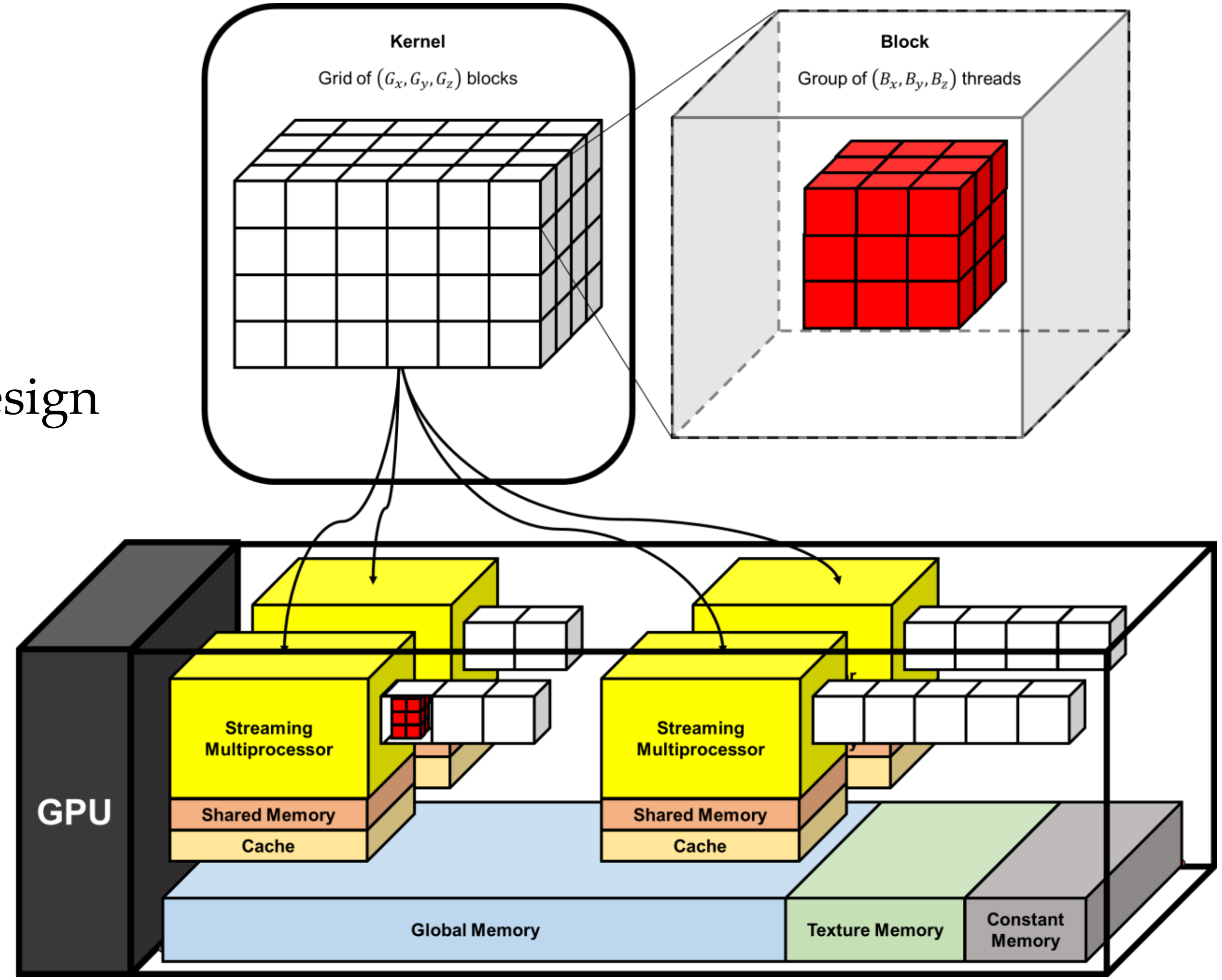
- **<u>A</u> <u>T</u>**oroidal **<u>L</u>**HC **<u>A</u>**pparatu**<u>s</u>**
- One of the two general-purpose detectors at the LHC
- Given the 40 MHz collision rate, we must filter out unwanted events
- This is the purpose of the Trigger system
- High-Luminosity LHC Upgrade (~2028) increases event density
- Hence, more computationally expensive to process the events in the Trigger
- Higher computational load → more computing power, better optimization of the algorithms or **hardware acceleration**
- Possibility for accelerating parallel operations: GPUs



GPU Programming

GPUs and GPU Programming

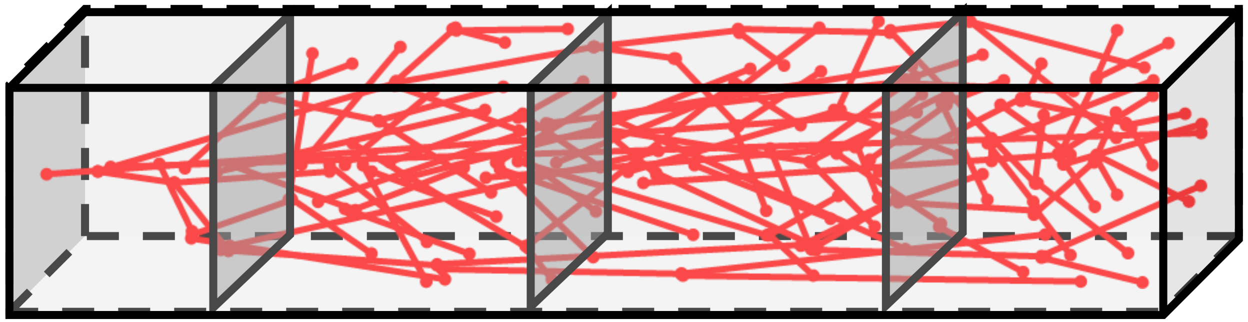
- <u>**G**</u>raphical <u>**P**</u>rocessing <u>**U**</u>nits
- Developed and designed to render 3D graphics
- Highly parallel operations → highly parallel design
- "SIMT": *Single Instruction – Multiple Threads*
- Branching is problematic
- Memory access latency issues become significant



Calorimeter Clustering Algorithms

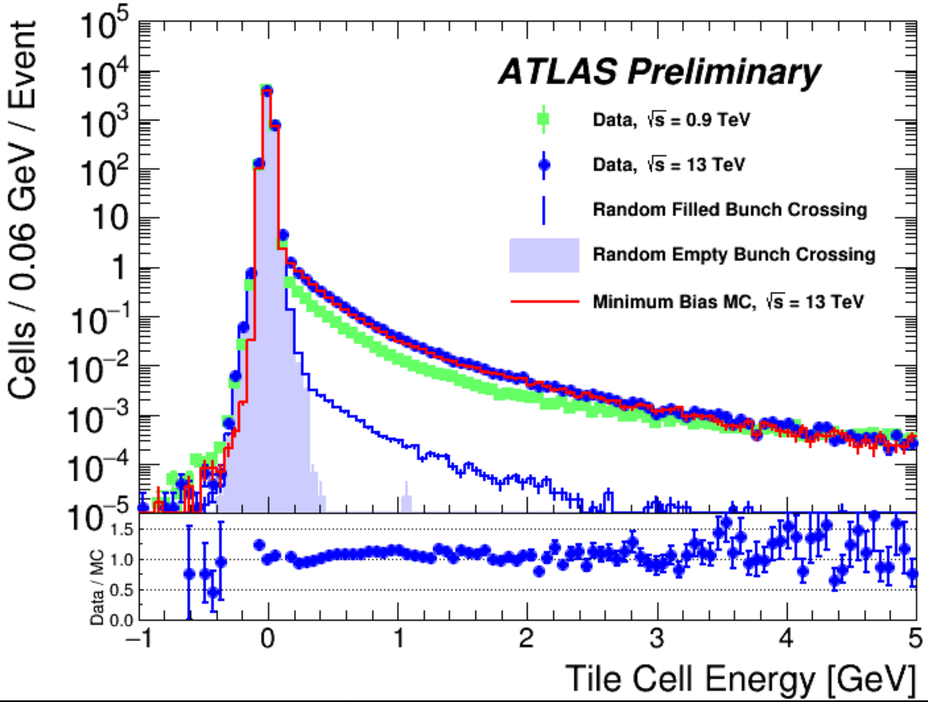
Calorimeter Reconstruction Algorithms

- Reconstruction of showers generated by outgoing particles



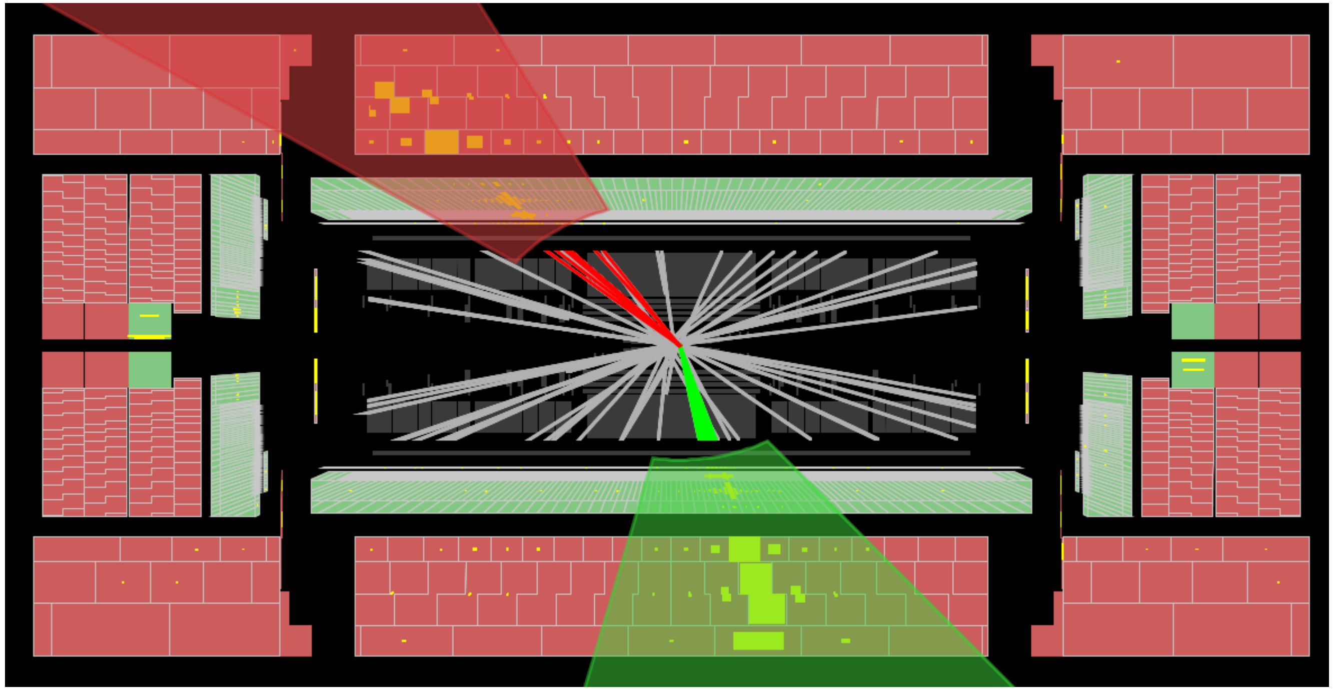
- Showers deposit their energy in a finite region of space: a calorimeter cell

- Two main sources of noise: electronic read-out and pile-up



Calorimeter Reconstruction Algorithms

- In general, several particles per event → reconstruction algorithms are clustering algorithms



- Topological Clustering grows the clusters according to the signal-to-noise ratio of energy measurements (by classifying cells as seed, growing, terminal or invalid according to thresholds)

- Topo-Automaton Clustering is a GPU-friendly variant that relies on a cellular automaton

Topo-Automaton Clustering

Specify T_{seed}, T_{grow}, $T_{terminal}$, with $T_{seed} > T_{grow} \geq T_{terminal}$

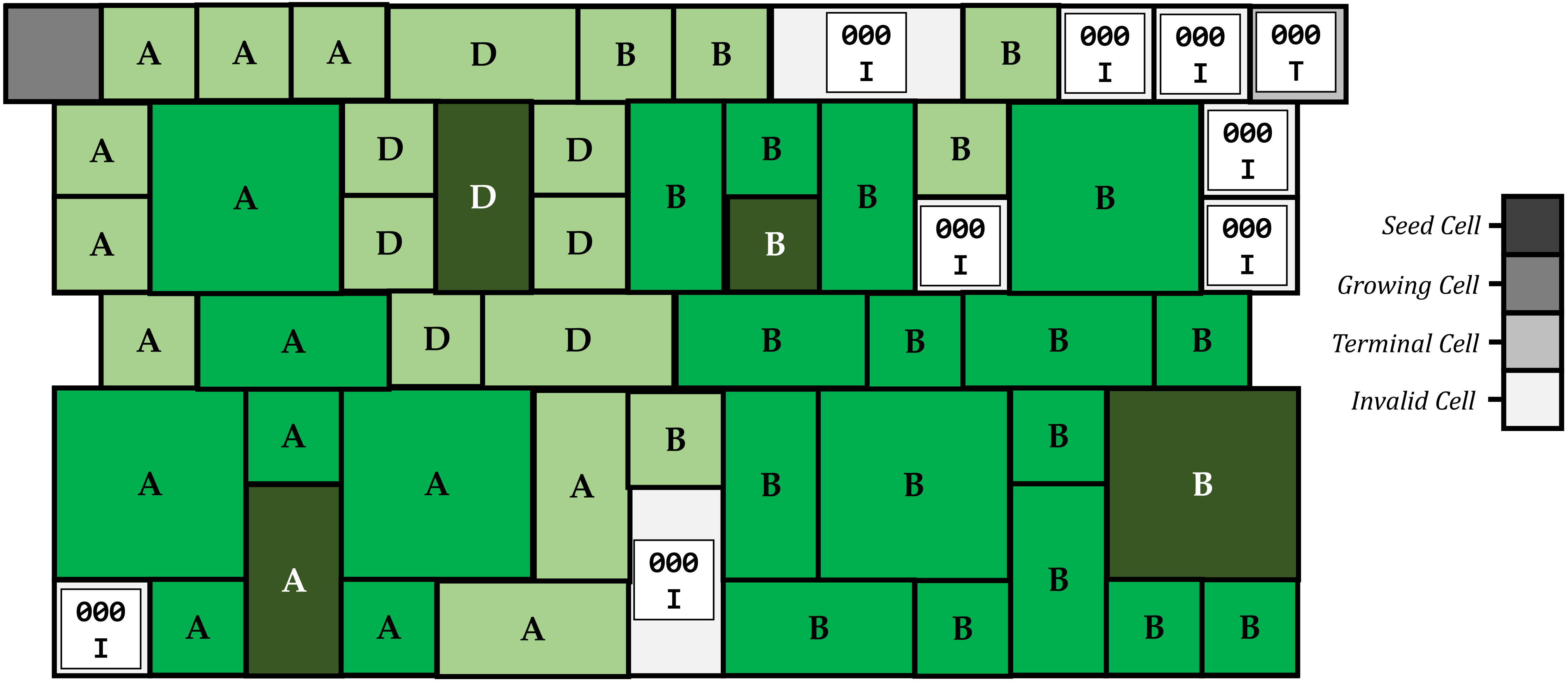
1. Classify each cell by the signal-to-noise ratio (SNR) of the deposited energy:
   - $SNR > T_{seed}$ → seed cell
   - $T_{seed} \geq SNR > T_{grow}$ → growing cell
   - $T_{grow} \geq SNR > T_{terminal}$ → terminal cell
   - $T_{terminal} \geq SNR$ → invalid cell
2. Generate a unique tag for every seed cell that reflects the ordering of the seed cell SNR, assign special "unassigned tags" to all others that compares lower than the unique tags (may use different values to distinguish growing, terminal and invalid)
3. Create the list of **eligible cell pairs** with the pairs of neighbours where one is at least growing (***cell G***) and the other is not invalid (***cell N***)
4. Repeat until there are no changes in the cell tags:

   For every (***cell G***, ***cell N***) pair in the list of **eligible cell pairs***:*

   Decrement the reverse propagation counter from the tag of ***cell N*** to get the propagated tag

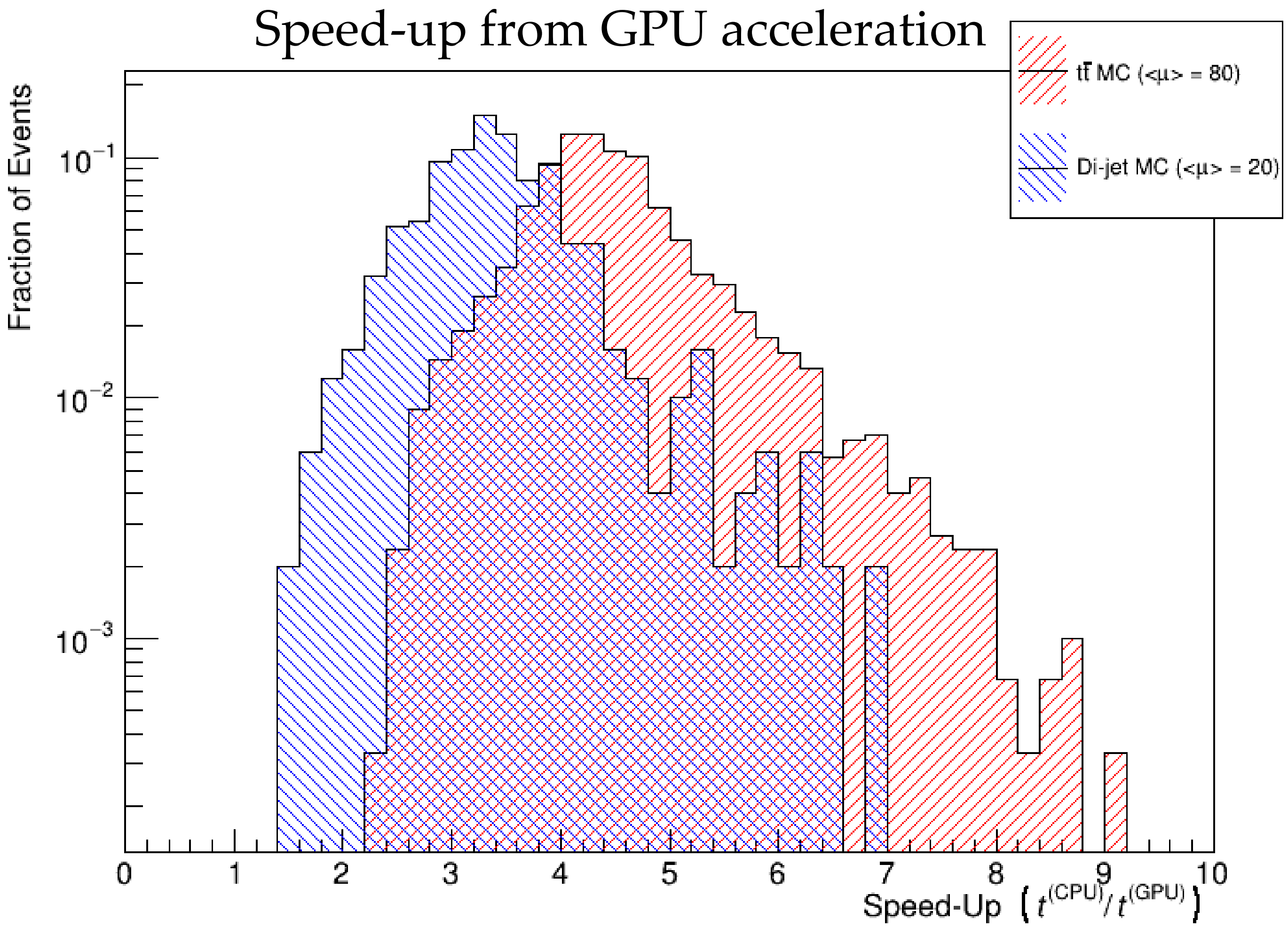
   If the propagated tag is larger than the tag of ***cell G***,
   - Assign the propagated tag to ***cell G***
   - If the tag of ***cell G*** was not the unassigned tag, merge the two protoclusters in question (keeping the counters)
5. For every pair of neighbouring cells where at least one is at least growing (***cell G***) and the other is not invalid (***cell N***):
   - Assign to the tag of ***cell N*** the maximum between its current tag and the tag of ***cell G***

Topo-Automaton Clustering



Preliminary Results

Results of the Validation



Speed-up of **~3.5** for di-jets, **~4.5** for $t\bar{t}$.

Less than 40% of the execution time is the algorithm

Results of Performance Comparison

Decomposition of the GPU event processing times

Step		t-$\bar{t}$ Events		Jet Events	
Preparing the Data	Input Data Conversion	2470 ± 729 µs (28%)	2249 ± 680 µs (25.5%)	2534 ± 715 µs (40.15%)	2313 ± 713 µs (36.65%)
	Input Data Transfer		221 ± 49 µs (2.5%)		221 ± 54 µs (3.5%)
Running the Algorithm	Signal-to-Noise Ratio Calculation	3212 ± 1456 µs (36.41%)	62 ± 8 µs (0.7%)	1918 ± 580 µs (30.39%)	59 ± 6 µs (0.93%)
	Pair Array Creation		137 ± 24 µs (1.56%)		85 ± 20 µs (1.35%)
	Cluster Growing		2776 ± 1352 µs (31.48%)		1588 ± 579 µs (25.16%)
	Final Cluster Properties Calculation		236 ± 72 µs (2.68%)		186 ± 11 µs (2.95%)
Packaging the Results	Output Data Transfer	3139 ± 800 µs (35.59%)	670 ± 400 µs (7.6%)	1859 ± 823 µs (29.46%)	666 ± 582 µs (10.56%)
	Output Data Conversion		2469 ± 400 µs (27.99%)		1193 ± 582 µs (18.9%)
Total		8824 ± 1602 µs		6316 ± 1443 µs	

50% to 55% of time spent in trigger ↔ GPU data structures
10% to 15% spent in CPU ↔ GPU data transfers

Ongoing and Future Work

- Restructure the code to decouple data conversion and transfer from actual work on the GPU (mostly done!)
- Integrate with further GPU-based calorimeter reconstruction, namely cluster splitting
- Measure speed-up and throughput in a multi-threaded context (preliminary results show relatively linear scaling)
- Understand the remaining differences between the terminal cell assignment of the algorithms
- Test compatibility of the GPU implementation with a realistic trigger chain
  (ensure it works when there are other algorithms using the GPU)

Thank you for your attention!

Backup Slides

The ATLAS Trigger

- ATLAS Trigger will be upgraded due to the High-Luminosity LHC Upgrade
- Phase I already implemented, Phase II will be done until Run 4
- Two stages:
  - Hardware-based (Level 1/Level 0)
  - Software-based (High-Level Trigger/Event Filter)
- Phase II upgrade increases event rate at the second stage from 100 kHz to 1 MHz
- Higher computational load → more computing power and/or better optimization of the algorithms
- Alternative to consider: hardware acceleration
- Possibility for accelerating parallel operations: GPUs

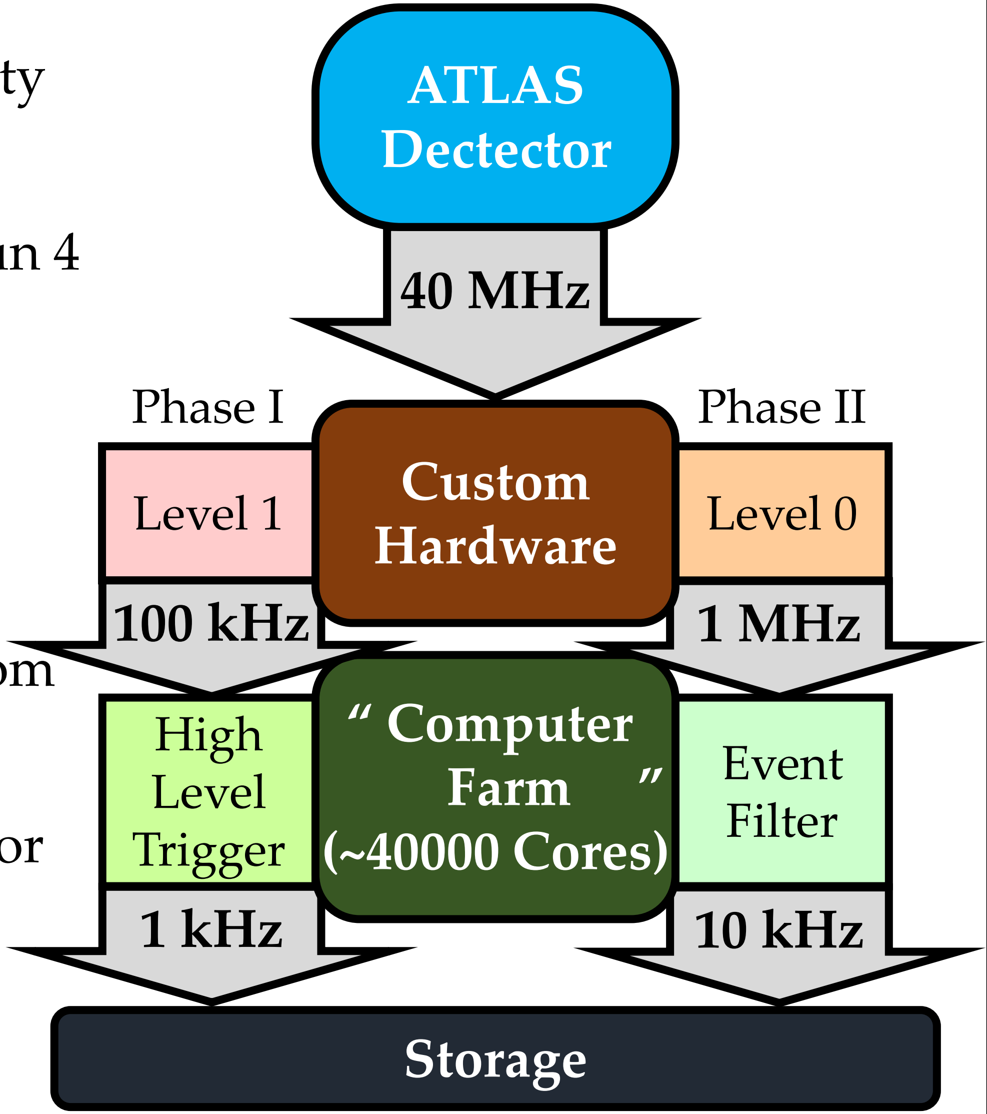


Diagram of the ATLAS Trigger System

Topological Clustering

Specify T_{seed}, T_{grow}, $T_{terminal}$, with $T_{seed} > T_{grow} \geq T_{terminal}$

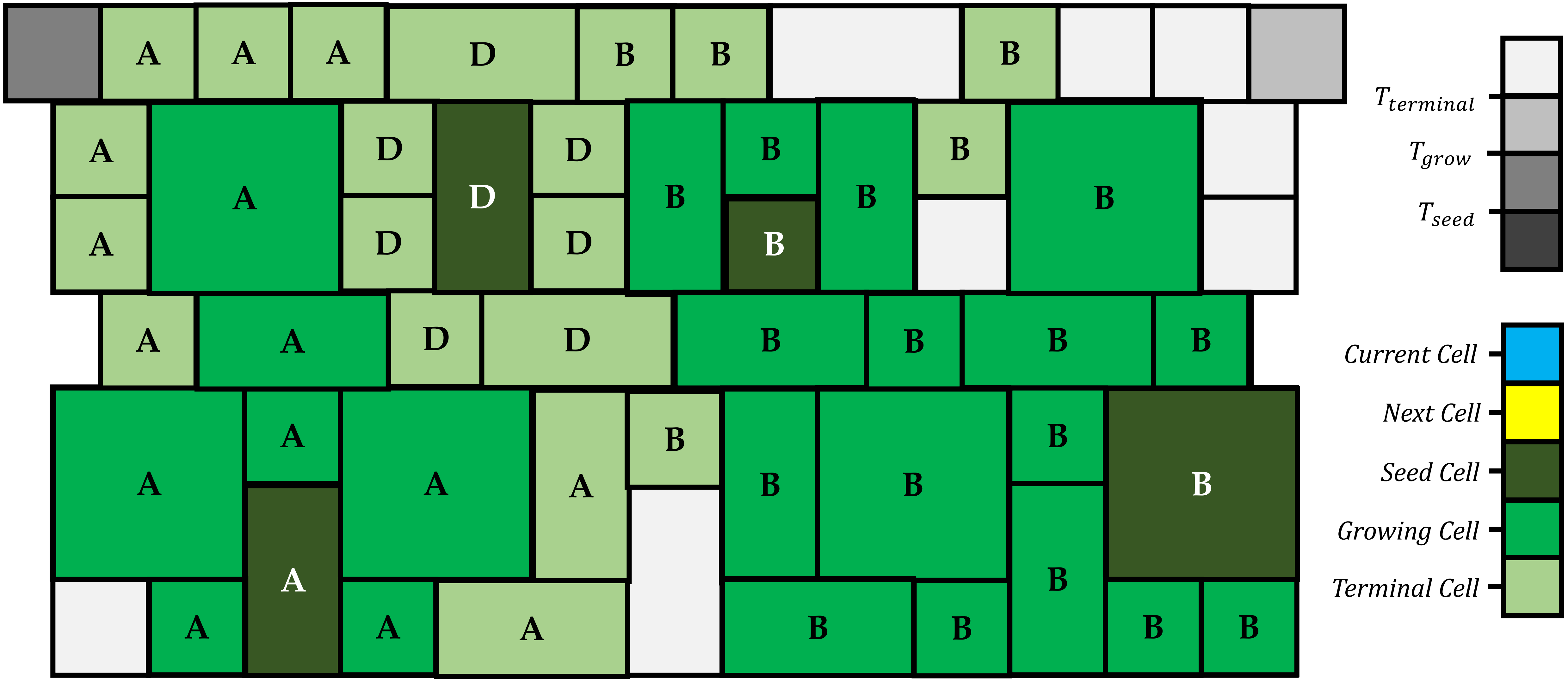
1. Classify each cell by the signal-to-noise ratio (SNR) of the deposited energy:
   - $SNR > T_{seed}$ → seed cell
   - $T_{seed} \geq SNR > T_{grow}$ → growing cell
   - $T_{grow} \geq SNR > T_{terminal}$ → terminal cell
   - $T_{terminal} \geq SNR$ → invalid cell
2. All seed cells are the origin of a cluster, add all seed cells sorted by their SNR to the **current cell list**, also create an empty **next cell list**
3. While the **current cell list** is not empty:
   1. For every cell in the **current cell list**:

      For every direct neighbour of the cell:

      a) If it does not belong to a cluster,
         - Add it to the current cell's cluster
         - If it is at least a growing cell, add it to the **next cell list**

      b) Else, if it already belongs to a cluster and it is at least a growing cell, merge the two clusters
   2. The cells in the **next cell list** become the **current cells** for the next iteration

Topological Clustering



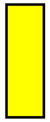
Topological Clustering on GPUs

- Previous demonstrations of porting Topological Clustering to GPUs ([1] and [2])
- Terminal cell attribution relies on serial behaviour and clusters as lists of cells is not GPU-friendly
- Solution: use a cellular automaton, propagate tags that identify each cell as belonging to a cluster
- Hence, Topo-Automaton Clustering

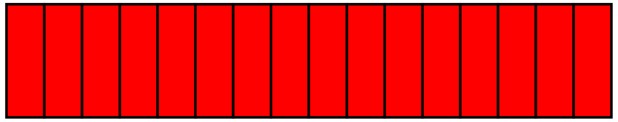
[1] – ATLAS Collaboration. “Multi-threaded algorithms for GPGPU in the ATLAS High Level Trigger”, DOI: 10.1088/1742-6596/898/3/032003

[2] – A. T. Delgado and D. Emeliyanov. “ATLAS trigger algorithms for general purpose graphics processor units”, DOI: 0.1109/NSSMIC.2016.8069670

Anatomy of a Cluster Tag



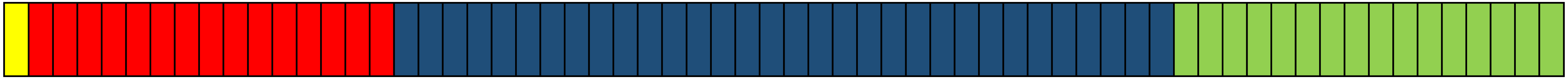
High bit to distinguish valid tags from terminal and growing cells



16 bit counter (2^{16} – #propagations)

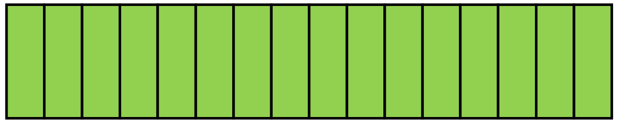
Assumptions:

- Less than $2^{16} = 65536$ clusters
- Less than 2^{16} propagation steps



SNR (highest bit is 0 since it's in absolute value)

Index of the cluster



Current Implementation

- Simple, linear structure that interfaces with GPU directly
- At the beginning of the run, send calorimeter geometry and noise to the GPU (approximation!)
- Three stages when processing each event:
  - *Preparing the data*: read and send deposited energy to the GPU
  - *Running the algorithm*:
    - Signal-to-Noise Calculation
    - Pair Creation
    - Cluster Growing
    - Final Cluster Properties Calculation
  - *Packaging the results*: get tag and cluster information from the GPU

Current Implementation