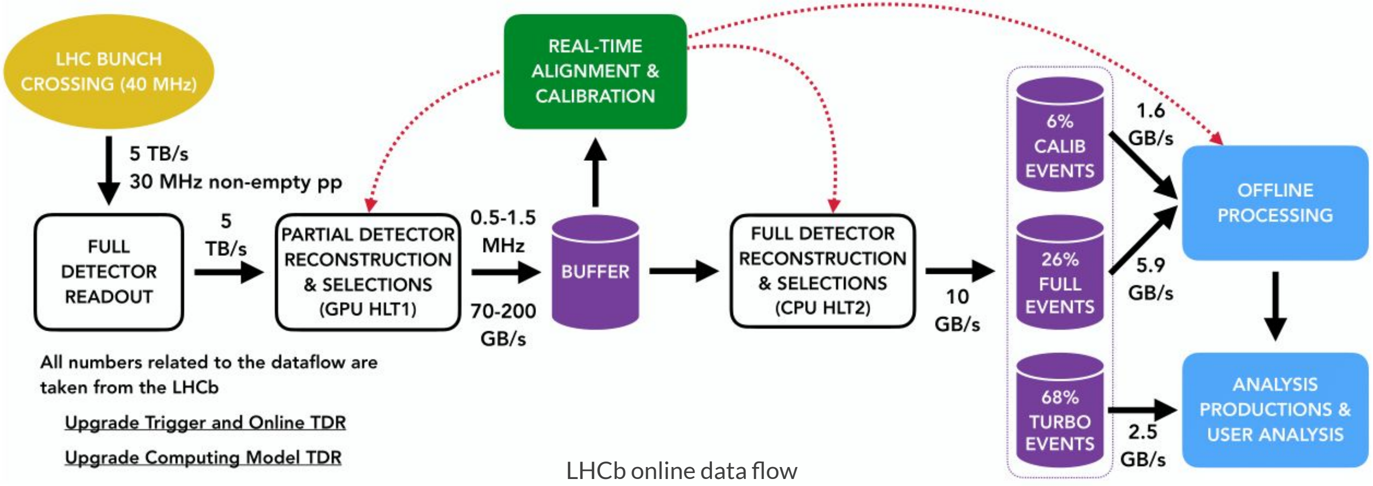
Performance Study of Alternative Calorimeter Clustering Solutions for Allen in LHCb

Núria Valls Canudas
Smart Society Research Group - La Salle URL Barcelona
On behalf of the LHCb Real Time Analysis Project

26th International Conference on Computing in High-Energy and Nuclear Physics
9th May 2023

1. The LHCb Trigger System

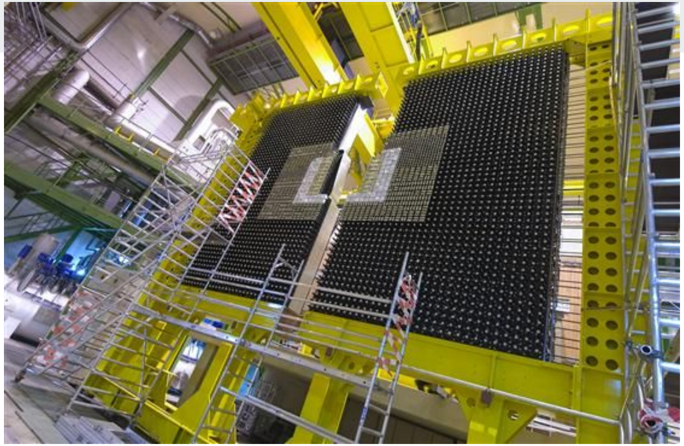
- The data flow generated from the LHCb detector currently reaches **5 TB/s**.
- Before storage, this rate is reduced by a factor 400 with the **trigger system**.
- **Real Time Analysis** approach: full event reconstruction and selection of specific signals of interest enabled by a quasi-real-time alignment and calibration.

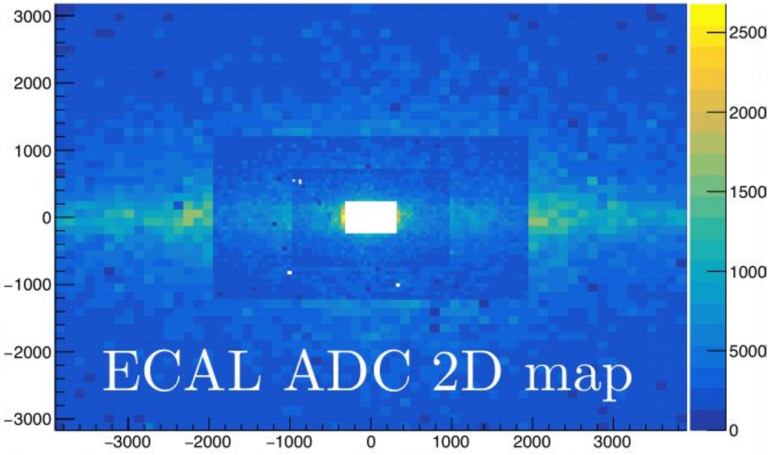


LHCb online data flow
LHCB-FIGURE-2020-016 (2020)

2. ECAL and reconstruction

- The **electromagnetic calorimeter** (ECAL) is used for photon and electron identification.
- Makes high precision measurements of position and energy deposited.
- 2D grid of Shashlik modules with three active regions:
    - Inner → 4x4 cm² cell size
    - Middle → 6x6 cm² cell size
    - Outer → 12x12 cm² cell size
- Output data: Grid of "digits" → energy deposits (MeV) on each calorimeter cell.
- Reconstruction: Cluster the digit deposits from the same particle (typically 3x3).





ECAL detector and ECAL ADC 2D map
LHCb-TALK-2022-163

2. ECAL and reconstruction in HLT2

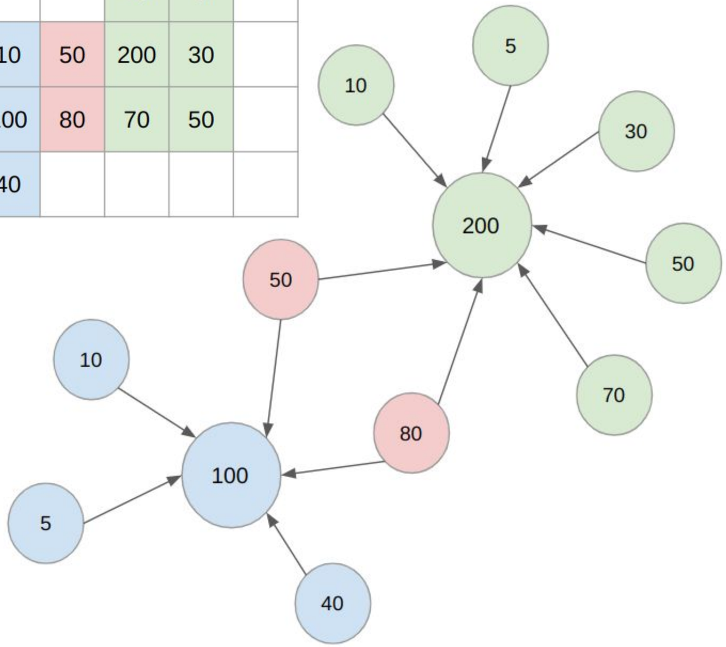
Graph Clustering is the algorithm that performs the full ECAL reconstruction in **HLT2**:

1. Sort the event digits by decreasing energy.
2. Insert digits into the graph under some rules.
3. Get the connected components of the graph.
4. Analyse each connected component to build the clusters:
    a. Resolve the **overlapping energy**.
    b. Output the reconstructed clusters.

C++ algorithm

Run in CPU

			10	5	
	10	50	200	30	
5	100	80	70	50	
	40				



An example of two clusters with overlapping cells
LHCb-DP-2022-003

3. ECAL and reconstruction in HLT1

A **simplified version** of the ECAL reconstruction was included in the HLT1 sequence for PID:

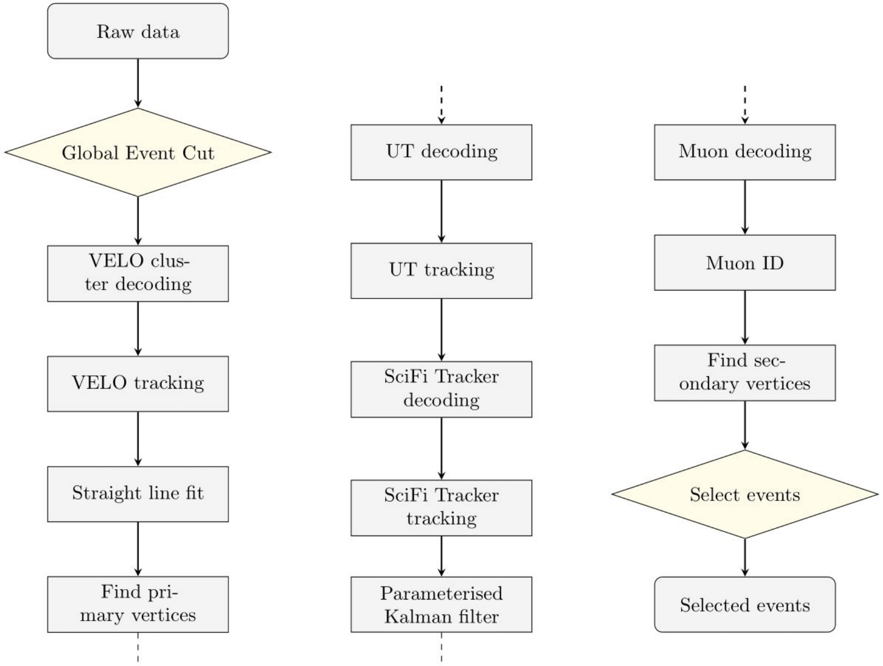
1. Search for local maxima cluster seeds.
2. Build a cluster with the 3x3 neighbors cells around the seed.

⚠ Overlapping cells are not resolved.

⚠ Only neighbors above a threshold are added to the cluster.

CUDA algorithm

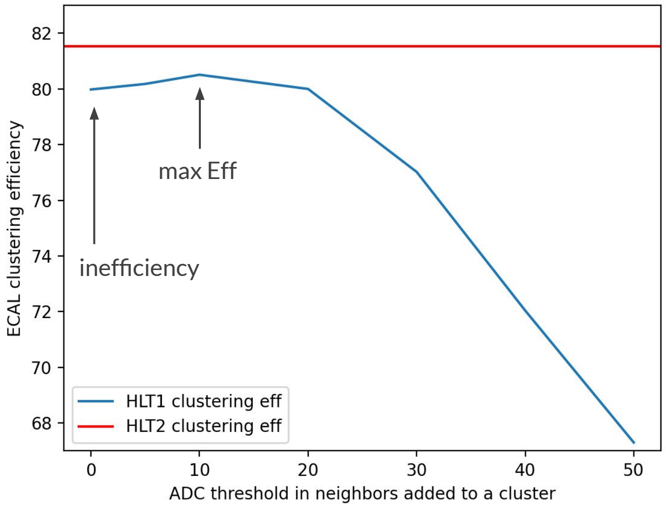
Run in GPU



Baseline HLT1 sequence
Updated from CERN-LHCC-2020-006 (2020)

4. Motivation for an improvement

- The simplified ECAL clustering needs to fit in the HLT1 throughput requirements →**fast** algorithm
- Current clustering efficiency is lower than the HLT2 algorithm → the **more efficient**, better selections and better trigger decisions.



- With the current approach, maximum efficiency is achieved excluding the neighbors with less than 10 ADC.

  → Symptom that shows the need of resolving the overlap cases.

- There is still room for improvement to reach the HLT2 efficiency.

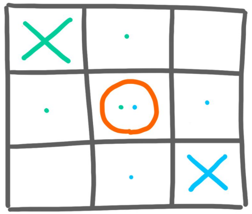
5. Proposal: first approach (A1)

Base idea: identify the cells that have overlap and correct the energy of the clusters accordingly.

1. Seed identification: local maxima
2. **Overlap cells identification**:
   - For each digit, count the number of neighbors that are tagged as a seed.
   - If there are two seeds, account an **overlap cell** with the two seed IDs.
3. Build clusters: using 3x3 neighbors
4. **Correct cluster energy**:
   - For each overlap cell, compute the energy correction as:

$$correction_{cluster1} = E_{overlap}\frac{E_{cluster2}}{E_{cluster1}+E_{cluster2}}$$

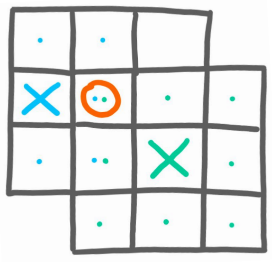
   - Subtract the respective energy fraction from the two tagged clusters



5. Proposal: second approach (A2)

Base idea: make a pre-calculation of the energy correction due to overlap before building the clusters.

1. Seed identification: local maxima
2. **Pre-calculate energy correction for each seed:**
   - For each seed, look at its neighbors.
   - For each neighbor, count the number of seeds around it
   - If there is a second seed, accumulate the correction for that cluster as:

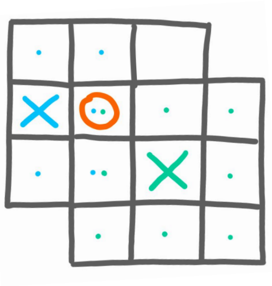


$$correction_{cluster1} = E_{overlap} \frac{E_{seed2}}{E_{seed1} + E_{seed2}}$$

3. Build clusters: using 3x3 neighbors and **subtracting the energy correction**

5. Proposal: second approach variation (A2.1)

Base idea: make a pre-calculation of the energy correction due to overlap before building the clusters.



1. Seed identification:
   - local maxima and **store the accumulated energy from the neighbors**
2. **Pre-calculate energy correction for each seed:**
   - For each seed, look at its neighbors.
   - For each neighbor, count the number of seeds around it
   - If there is a second seed, accumulate the correction for that cluster using the total energy of the cluster:

$$correction_{cluster1} = E_{overlap}\frac{E_{cluster2}}{E_{cluster1}+E_{cluster2}}$$

3. Build clusters: using 3x3 neighbors and **subtracting the energy correction**

6. Results

Comparing performance with the original calo clustering in HLT1 and the HLT2 algorithm:

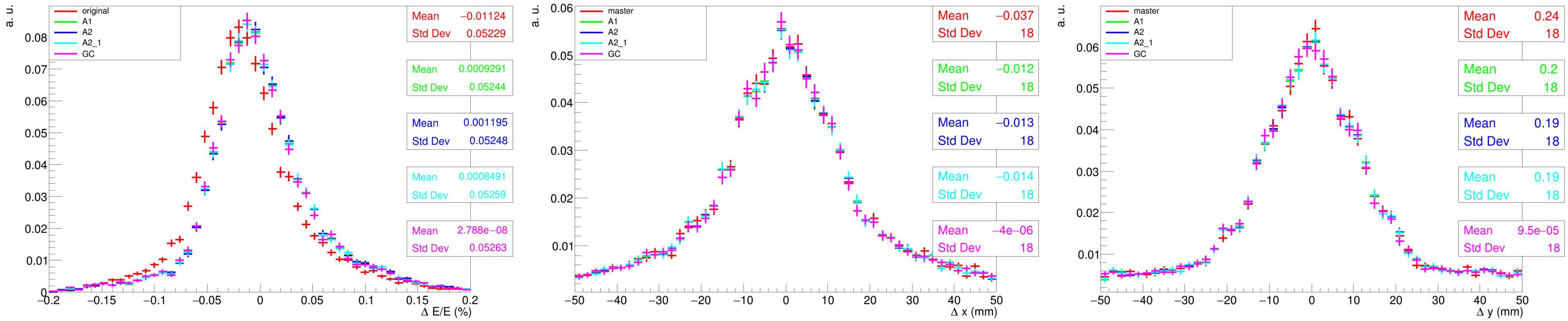
Graph Clustering efficiency:
81.54±0.28

Approach	Efficiency (%)	Throughput (events/s)
Original	80.51 ± 0.29	99911.03
A1	81.17 ± 0.29	96515.43
A2	81.04 ± 0.29	97383.57
A2.1	81.32 ± 0.29	97141.25

- Efficiency:percentage of reconstructed clusters vs reconstructible clusters using ~50k simulation events in LHCb Upgrade I conditions.
- Throughput: max of 20 executions in GPU NVIDIA GeForce RTX 2080 Ti.

6. Results

Comparing energy and position resolution with the Graph Clustering as a baseline:



~100k simulation events from $B^0 \rightarrow K^*$gamma: Energy resolution, X position resolution and Y position resolution without corrections.

⟶ No significant improvement is seen.

7. Conclusions

- We have started to study three alternatives that implement the calorimeter shower overlap resolution in the Allen framework.
- The approaches trade off between efficiency and throughput, however, no significant improvements in resolution.
  - Approach A2_1 reaches an efficiency comparable to Graph Clustering with a moderate throughput decrease.
- WIP:
  - Check the impact on the energy resolution for low energy photons.
  - Study the impact of efficiency increase to physics performance.

⟶This work aims to set a starting point to the study of the implications of the shower overlap in the HLT1 calorimeter clustering.

Thank you!

Any questions?

Núria Valls Canudas
Smart Society Research Group - La Salle URL Barcelona
On behalf of the LHCb Real Time Analysis Project

26th International Conference on Computing in High-Energy and Nuclear Physics
9th May 2023

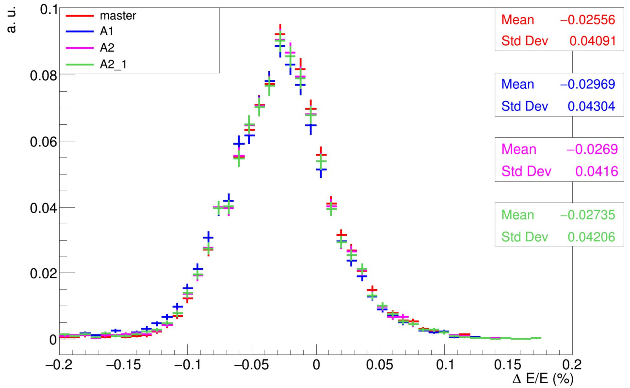
Backup

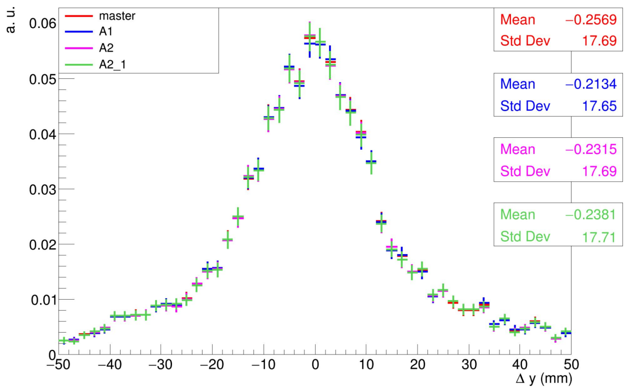
Núria Valls Canudas
Smart Society Research Group - La Salle URL Barcelona
On behalf of the LHCb Real Time Analysis Project

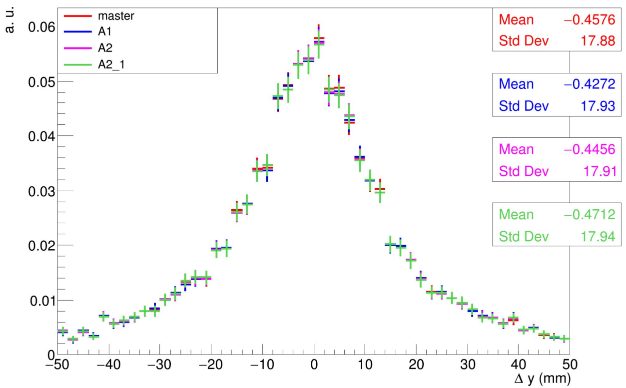
26th International Conference on Computing in High-Energy and Nuclear Physics
9th May 2023

Pi0 resolution plots

~100k simulation events from B0 → π +π −π 0: Energy resolution, X position resolution, Y position resolution.







Graph Clustering algorithm steps

Algorithm steps:

1. **Sort** the event digits by decreasing energy.
2. **Insert** digits into the graph.
3. Get the **connected components** of the graph.
4. **Analyse** each connected component to build the clusters.

Graph Clustering algorithm steps

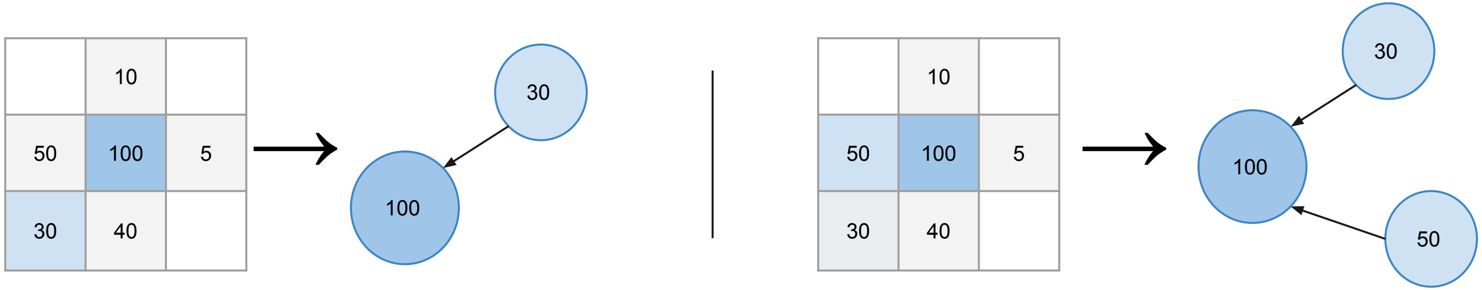
1. Sorting

- It is needed to make sure the seeds of clusters are inserted in the graph before its neighbor digits.
- 50 MeV is the minimum energy of a digit to be considered a seed.
- Only digits above 50 MeV are sorted by decreasing energy value.

Graph Clustering algorithm steps

2. Insertion

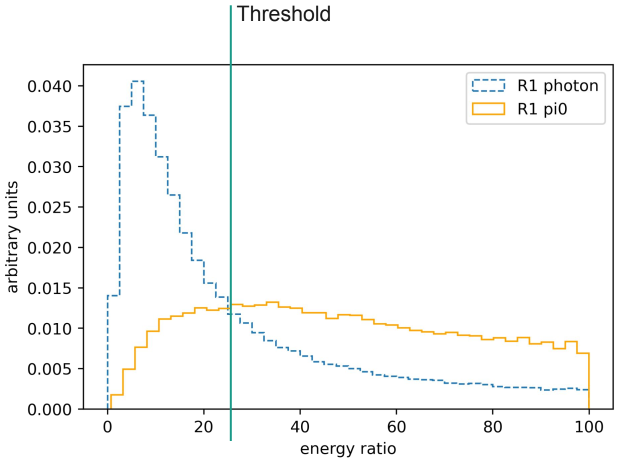
- Start with the highest energy digits (only possible seeds):
  - If it is already inserted in the graph it is already part of a cluster → cannot be a cluster seed.
  - It not, if it is a local maxima → is a cluster seed.
    - Insert all the distance 1 neighbors to the graph and link them to the seed with a directed edge.

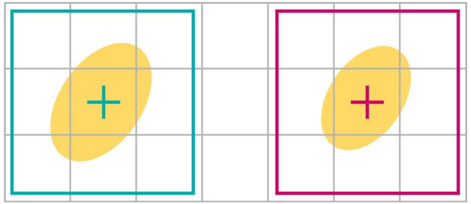


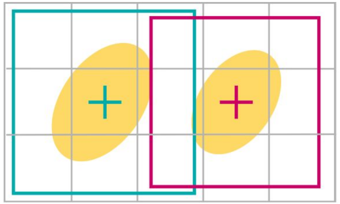
Graph Clustering algorithm steps

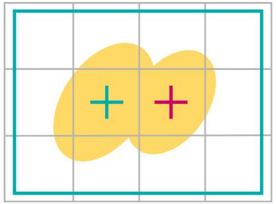
2. Insertion: particular case

- Neutral pions decay into two photons before reaching the calorimeter.
- Merge π^0s are reconstructed as a single 3x3 cluster (by default), leaving significant energy deposits out.
    - To avoid this, potential merged π^0s are filtered by the energy ratio between the seed and the second most energetic deposit (R1).
    - This clusters are expanded to the neighbor cells of the second seed.




Separable photons, no overlap


Resolved π^0

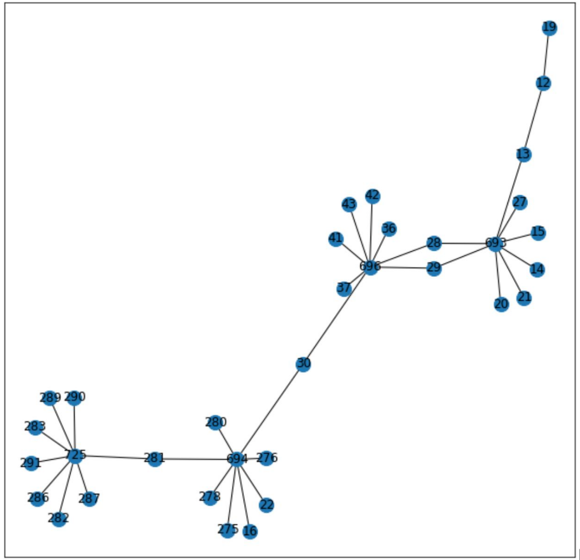

Merged π^0

Graph Clustering algorithm steps

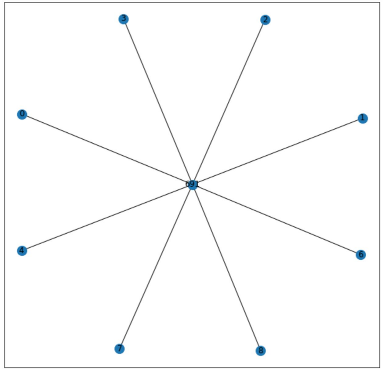
3. Connected Components

- Given a directed graph, a weakly connected component (WCC) is a subgraph of the original graph where all vertices are connected to each other by some path, ignoring the direction of edges.
- ~60% of WCCs are already isolated clusters itself

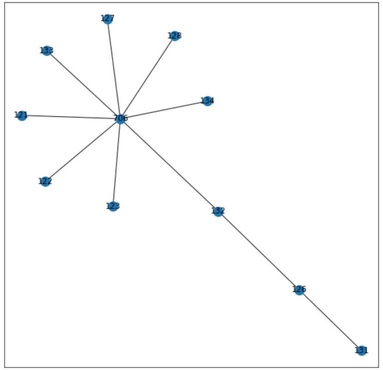
"Big" CC



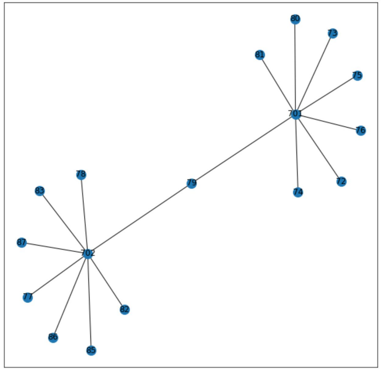
Isolated cluster



Cluster overlapping with small residual cluster



Two clusters overlapping in one cell



Graph Clustering algorithm steps

4. Analysis of Connected Components

- Identifies overlap cells and calculates which fraction of its energy is assigned to each cluster.
- Uses an estimation based on the total energy of each cluster:

$$fraction_{cluster1} = \frac{E_{cluster1}}{\sum_{i=0}^{N} E_{cluster\ i}}$$

- Although it takes ~3 iterations to fully converge, with only 1 iteration the efficiencies do not change significantly and execution time is minimized.
- At the end of this step, the list of clusters is already completed with the information of: `seed digit`, `total energy`, list of `digits in the cluster` with `id`, `energy` and `fraction`.