Topo Clustering and EDM4hep

Brieuc François (CERN)

FCC software meeting

Oct. 30th, 2020

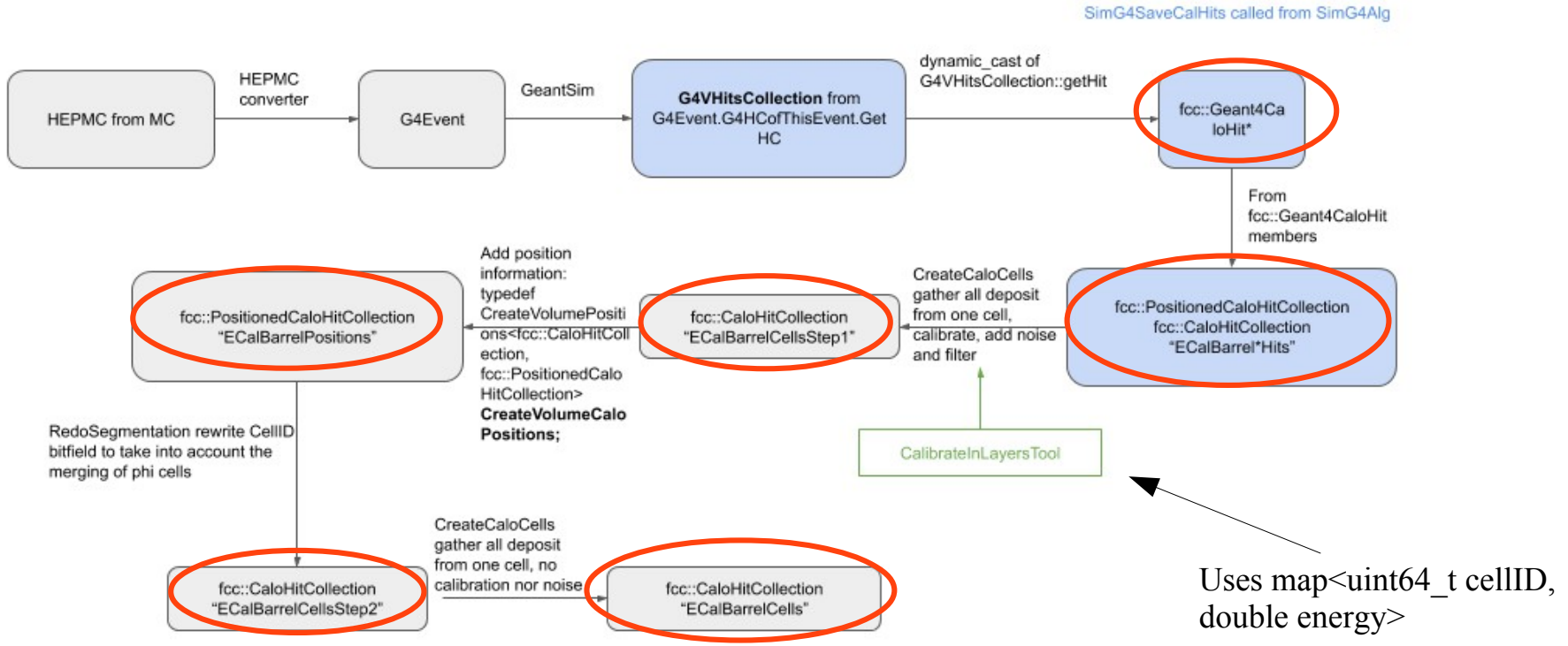
Introduction

- Starting thorough FCC-ee LAr ECAL optimization campaign
- Better to do any new development aligned with the 'long term' approach → EDM4Hep
- Sharing here my experience as a newcomer having a first look at ECAL reconstruction in FCCSW
- Running the calorimeter reconstruction in FCCSW
  - Useful technical (“howto's”) documentation from Jana: link
    - Everything works out of the box except the upstream material correction, you need to add this snippet to the provided config file and add it to the TopAlg

```
from Configurables import CreateCaloCells
createcellsBarrel = CreateCaloCells("CreateCaloCellsBarrel",
                                    doCellCalibration=False,
                                    addCellNoise=False, filterCellNoise=False)
createcellsBarrel.hits.Path="ECalBarrelHits"
createcellsBarrel.cells.Path="ECalBarrelCells"
```

Calorimeter reconstruction

- Workflow (probably not super accurate) of the python config producing fcc::CaloHitCollection used as input by clustering: runCaloSim.py
  - Highlighting where fcc::edm is used (some algorithm called several times)
    - Four classes to modify: SimG4SaveCalHits, CreateCaloCells, CreateVolumePositions, RedoSegmentation



Topological Clustering

- The topological clustering works as follows (simplified)
  - Find all cells that have a signal to noise ratio (S/N) above a certain threshold (default = 4σ)
  - Add to all these seeds the neighboring cells that have a S/N above another (milder) threshold (default = 2σ), iteratively i.e. these neighboring cells becomes seed for a new round
  - Add cells considered in the previous step but that were not exceeding the 'milder' threshold, can again choose a threshold but the default is 'all cells added'
- header, config example

Topological Clustering in FCCSW

- **Input fcc::edm dependent**
  - map<uint64_t cellID, double energy> created by CaloTopoClusterInputTool based on fcc::CaloHitCollection – from all six calorimeters
- **Outputs:** fcc::CaloClusterCollection (uses fcc::CaloCluster **internally)** and fcc::CaloHitCollection
- **Tools relying on fcc::edm:** all the one inheriting from ICellPositionsTool e.g. CellPositionsECalBarrelTool (tools to look-up the cells positions by cellID, not sure why it is needed, maybe to have the eta segmentation)
- Tools free from fcc::edm: the one inheriting from ICaloReadNeighboursMap and ICaloReadCellNoiseMap e.g. ConstNoiseTool, NoiseCaloCellsFlatTool
- Comments
  - TopoCaloNeighbours uses as input a root file that contains a TTree with cellID and vec<neighboursCellID> produced by CreateFCChhCaloNeighbours, in a separated step
  - I think the only place where ordered map is necessary is for the seeds (order them by energy), could probably use unordered_map everywhere else

Migration to EDM4hep

- Would be nice to be able to test the algorithm during the migration (not just having it compile)
  - Create the needed dataformats in EDM4hep

```
# really need to expose some function f(Cellid) -> position
fcc::BareHit:
  cellId : unsigned long long
  energy : float
  time : float
  bits : unsigned

fcc::CaloHit:
  Description : "A calorimeter hit"
  Author : "C. Bernet, B. Hegner"
  Members:
   - fcc::BareHit core // contains basic hit information
```

```
fcc::Point:
  x : float
  y : float
  z : float


fcc::PositionedCaloHit:
  Description: "A calorimeter hit with its global position"
  Author: "J. Lingemann, B. Hegner"
  Members:
   - fcc::Point position // The global position
   - fcc::BareHit core // The hit
```

  - Bypass the lower-level collections by writing directly dummy edm4hep::caloHitCollection to a rootfile with write_events.cc and run the new topoClustering on it

- Problem: Having a simple python config running in the edm4hep branch seems not to be easy at the moment