Truth Matching

Rafayel Paremuzyan

- The objective
- Implementation
- Issues, current status and path forward to completion

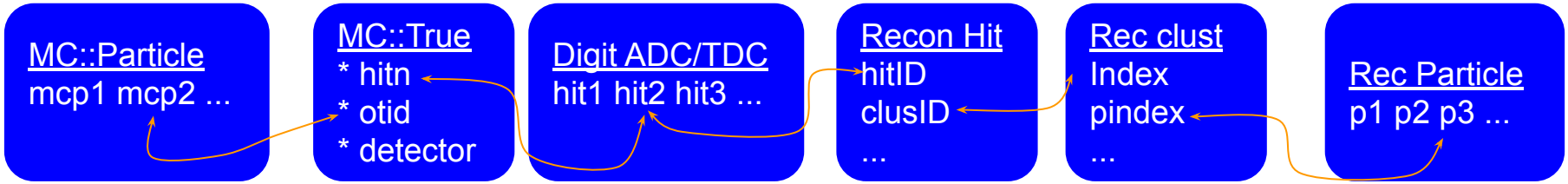
CLAS Collaboration meeting. March 2-5 2021

Objective

Rigorously (as much as possible) match the reconstructed particle to the True (MC particle)

No comparison of kinematics between generated and reconstructed particles in order to make a decision on the matching.

The key point in the matching is that in the MC::True bank the variable “hitn” shows the the index of the adc/tdc value in the corresponding digitized bank.
The MC::True bank has a variable “otid”, which shows the index of the original particle in the MC::Particle bank.



Two types of matching

- We have an MC particle, and want to find the corresponding Recon particle
  - Efficiency studies
  - Momentum, angular resolutions

```
MCPart_1 -> Rec_1
MCPart_2 -> Rec_2
...
MCPart_n -> Rec_n
```

The number of rows in this bank is equal to the number of MC::Particles

- We have a recon particle, and we want to know the MC::Particle that is responsible for the given Recon particle
  - Useful when there is a decay of the particle, and decay products are detected as different recon particles.

```
Rec_1 -> MCPart_1
Rec_2 -> MCPart_2
...
Rec_n -> MCPart_n
```

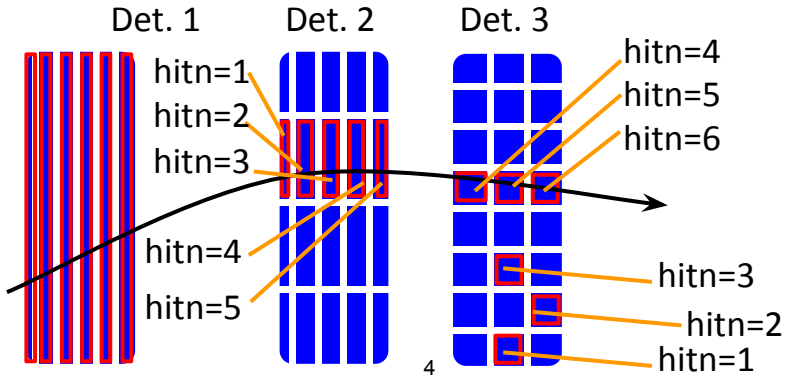
The number of rows in this bank is equal to the number of Rec::Particles

Getting true hits

```
// <Detector, <hitn, MCHit>>
Map<Byte, Map<Integer, MCHit>> mchits = getMCHits(event.getBank( "MC::True"),
mcp);
```

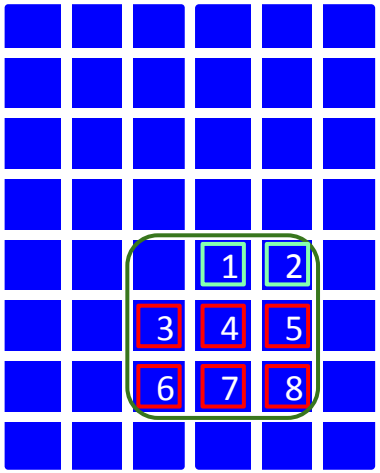
```
class MCHit {
    public int pid;       // MC particle id (pdg code)
    public int otid;      // id of the original (generated) particle that eventually caused the hit
    public int hitn;      // Hit id: it corresponds to the position of rec hits.
    public byte detector; // Detector code descriptor
}
```

For each detector get Map of MCHit objects.
MCHit tells which MCParticle is the original particle (otid), pid, detector and hitn (hit number in the given detector)
Note: here the hitn plays key role in the matching. The hitn should be equal to the adc index in the adc bank for the given particle.



Match clusters to MC Particles

The purpose: for each cluster we want to know which MC particle is responsible for this cluster.



It is possible that different hits from a cluster to be created from different MC particle
MC Particle #1
MC Particle #2

Examples:

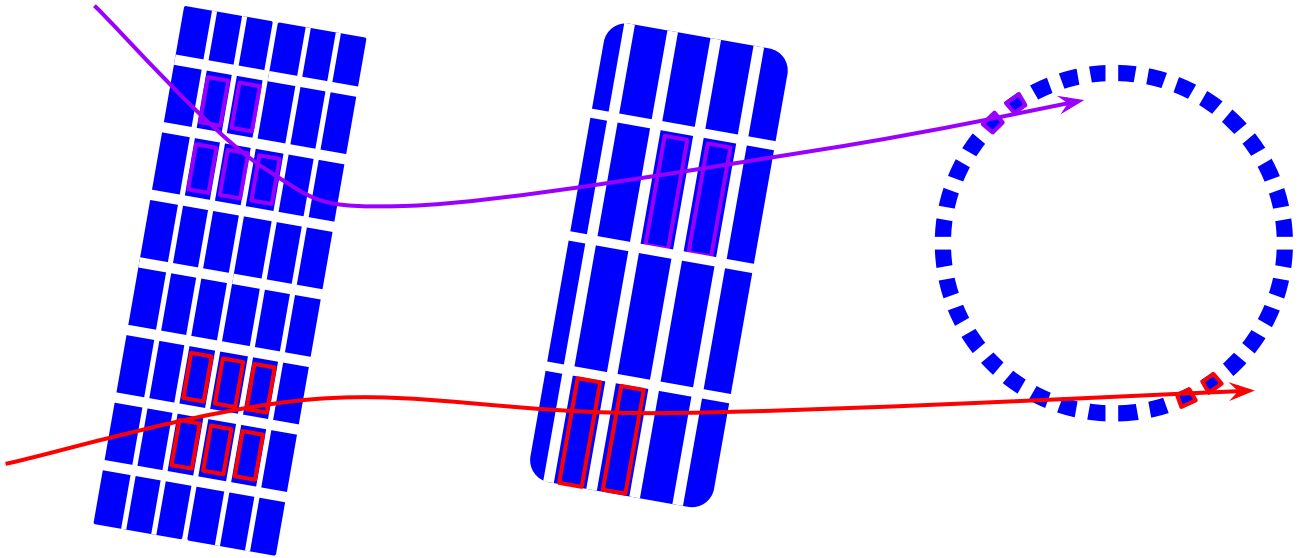
- Two particles hit the calorimeter close to each other and some hits of the cluster could be from one particle, and others from another particle.
- Some backscatter photon from detector material hit the same/next CTOF/CND channel where the other MCParticle hits.

So, which MC particle should be assigned to the given cluster?

Currently the MCParticle having highest number of hits in the cluster, is assigned as the MCParticle responsible for this cluster.

Each cluster object has it's associated MCParticle and Rec::Particle

Mapping clusters to MC and Rec Particles

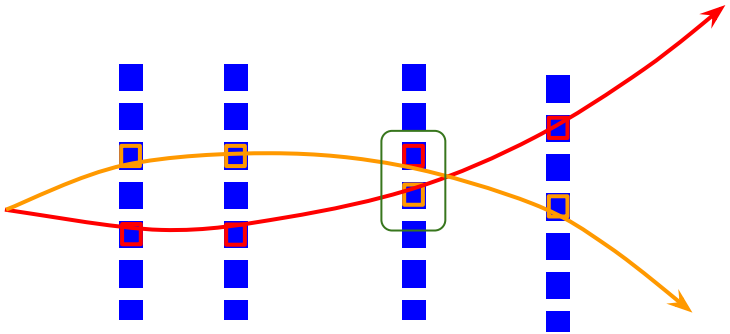


For each MC and Rec Particles we compiled the list of clusters.

Make MCRecMatch

At this point for a given MCParticle we have a list of RecCluters: Next step is to find out which Recon Particle those clusters point to. ***Is it a single Recon particle? Most of the time yes!, however strictly speaking Not necessarily...***

The one with highest number of clusters is assigned as the matched Recon particle to the given MCParticle



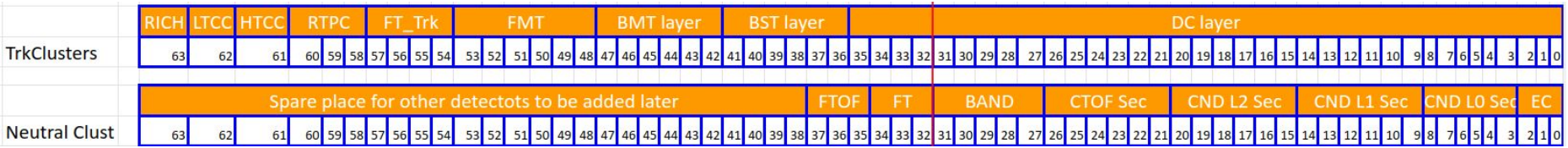
The information in the output bank


```
      "name": "MC::IsParticleMatched ",
       "group": 40,
       "item" : 5,
       "info": "MC Particle - Rec Track matching. ",
       "entries": [
            {"name":"mcTindex",      "type":"S",  "info":"MC Particle index, as in tid "},
            {"name":"pindex",        "type":"S",  "info":"REC particle index "},
            {"name":"MCLayersTrk",   "type":"L",  "info":"layers of tracker (mostly) detectors that a given MCParticle hit. Each layer
                                                          corresponds to a specific bit in this variable  "},
            {"name":"MCLayersNeut",  "type":"L",  "info":"layers of detectors (non tracker) that a givem MCParticle hit. Each layer
                                                          corresponds to a specific bit in this variable  "},
            {"name":"RecLayersTrk",  "type":"L",  "info":"layers of tracker detectors (mostly) that, the matched RecParticle picked hits
from"},
            {"name":"RecLayersNeut", "type":"L",  "info":"layers of detectors (mostly) that, the matched RecParticle picked hits from   "}


       "name": "MC::IsRecParticleMatched ",
       "group": 40,
       "item" : 6,
       "info": "Rec Particle - MC matching. ",
       "entries": [
            {"name":"pindex",        "type":"S",  "info":"REC particle index "},
            {"name":"mcTindex",      "type":"S",  "info":"MC Particle index, as in tid "},
            {"name":"RecLayersTrk",   "type":"L",  "info":"layers of tracker (mostly) detectors that a given Rec::Particle hit. Each layer
corresponds to a specific bit in this
variable"},
            {"name":"RecLayersNeut",  "type":"L",  "info":"layers of detectors (non tracker) that a givem Rec::Particle hit. Each layer
corresponds to a specific bit in this va
riable"},
            {"name":"MCLayersTrk",  "type":"L",  "info":"layers of tracker detectors (mostly) that, the matched MC::Particle picked hits
from"},
            {"name":"MCLayersNeut", "type":"L",  "info":"layers of detectors (mostly) that, the matched MC::Particle picked hits from   "}
       ]
   }
```


Jefferson Lab

Status words



Let say the MC particle deposited hits in ALL DC layers except a layers 2 and 3 from the 1st SL then the binary representation of the MCLayersTrk will be “11111111111111111111111111111001” (decimal: 68719476729).
Now the matched particle used none of first SL hits, and also didn’t used first and last hits from the third SL. Then the binary representation of RecLayersTrk will look like “11111111111111111011110111111000000: (decimal 68719341504)

Important: the non negative p-index doesn’t necessarily mean you should consider the corresponding recon particle as the matched particle. Te particle with a given p-index is just the recon particle among other recon particles, which has the maximum number of matched hits to this MC particle, however the number of matched hits can be for example 1. In this case you most probably will not consider as the matched particle.

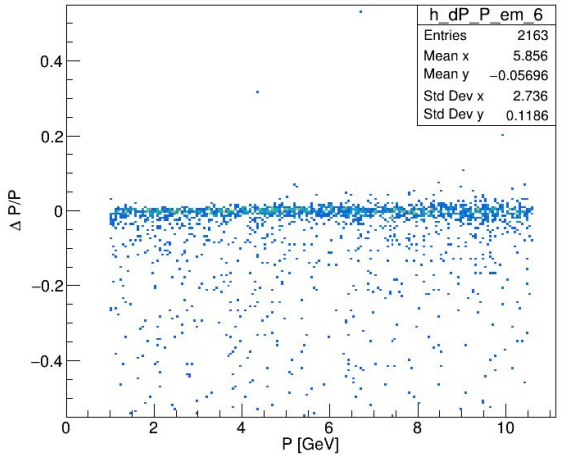
There are functions that allows to check some basic properties of the matched particle using the status word

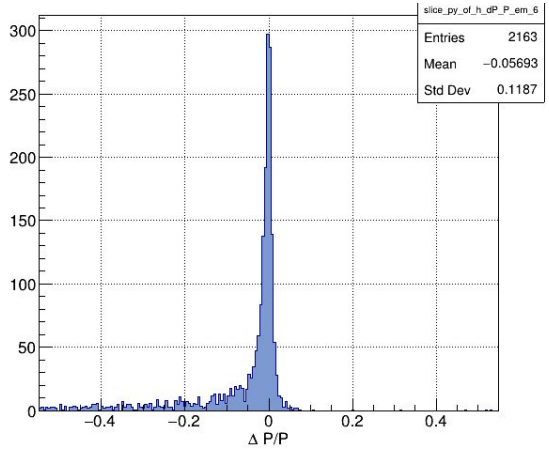
- `public Integer CountNSetBits(Long word, short bit1, short bit2) {`
- `public Boolean CheckDCAcceptance(Long word, short nMinSL, short nMinLayerPerSL) {`

Issues and the current status

- This doesn’t work with bgr merging: inserted hits brake relation between True and dgtzd hits
  - Will be fixed soon when gemc will output directly hipo
- The DC inefficiency introduced in GEMC, brakes the True<-> dgtzd hit relationship for about 2% of hits
  - This will be fixed in the with the hipo output in gemc
- Truth matching is added in the COATJAVA release 6.6.0.
  - In the release only the MC->Rec matching is done.
    - The Rec2MC matching is now developed and being tested in the `iss540-Recon2MCMatching`
- Only tracking detector are in the status word currently. Others will follow soon.

Some basic tests were done, but definitely it will be very useful to get feedback, bug reports from users.





Jefferson Lab