Track-Based Particle Flow

Outline:

- Introduction
- Details of the Algorithm
- Performance of Track-Based Particle Flow
- Conclusions and Outlook

Introduction

Goal for precision physics:

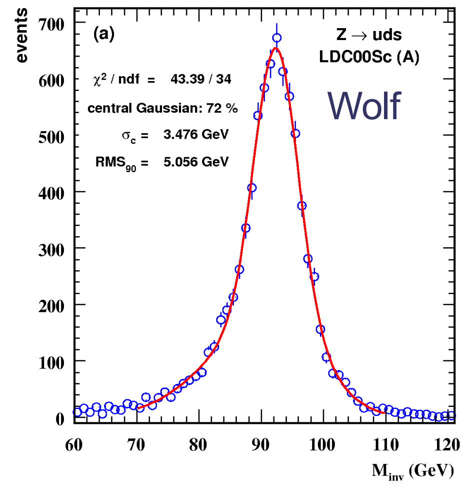
- Jet-Energy Resolution **0.30**/$\sqrt{E_{jj}}$ in hadronic decays of Z^0 and $W^{+/-}$
- → corresponds to a boson mass resolution of approx. $\Gamma_{Z/W}$
- 'Particle Flow Concept' is able to **reach** this goal

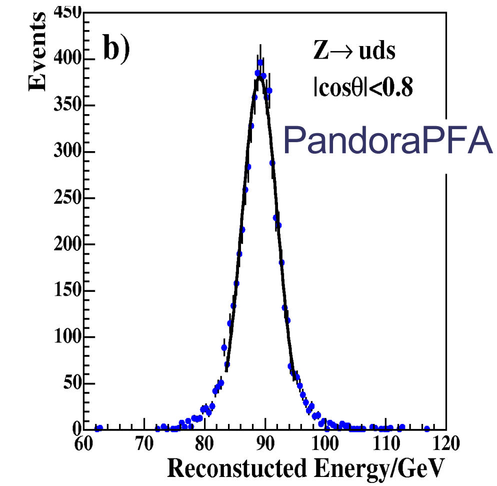
Particle Flow Algorithms for LDC (Marlin/MarlinReco):

1. Wolf: $\Delta E/E \approx$ **0.53**/$\sqrt{E}$ for $Z^0 \rightarrow$ uds for E_{Jet}= 45 GeV
2. PandoraPFA: $\Delta E/E \approx$ **0.30**/$\sqrt{E}$ for $Z^0 \rightarrow$ uds up to E_{Jet}= 100 GeV
3. TrackBased PFlow: $\Delta E/E \approx$ **0.41**/$\sqrt{E}$ for $Z^0 \rightarrow$ uds for E_{Jet}= 45 GeV

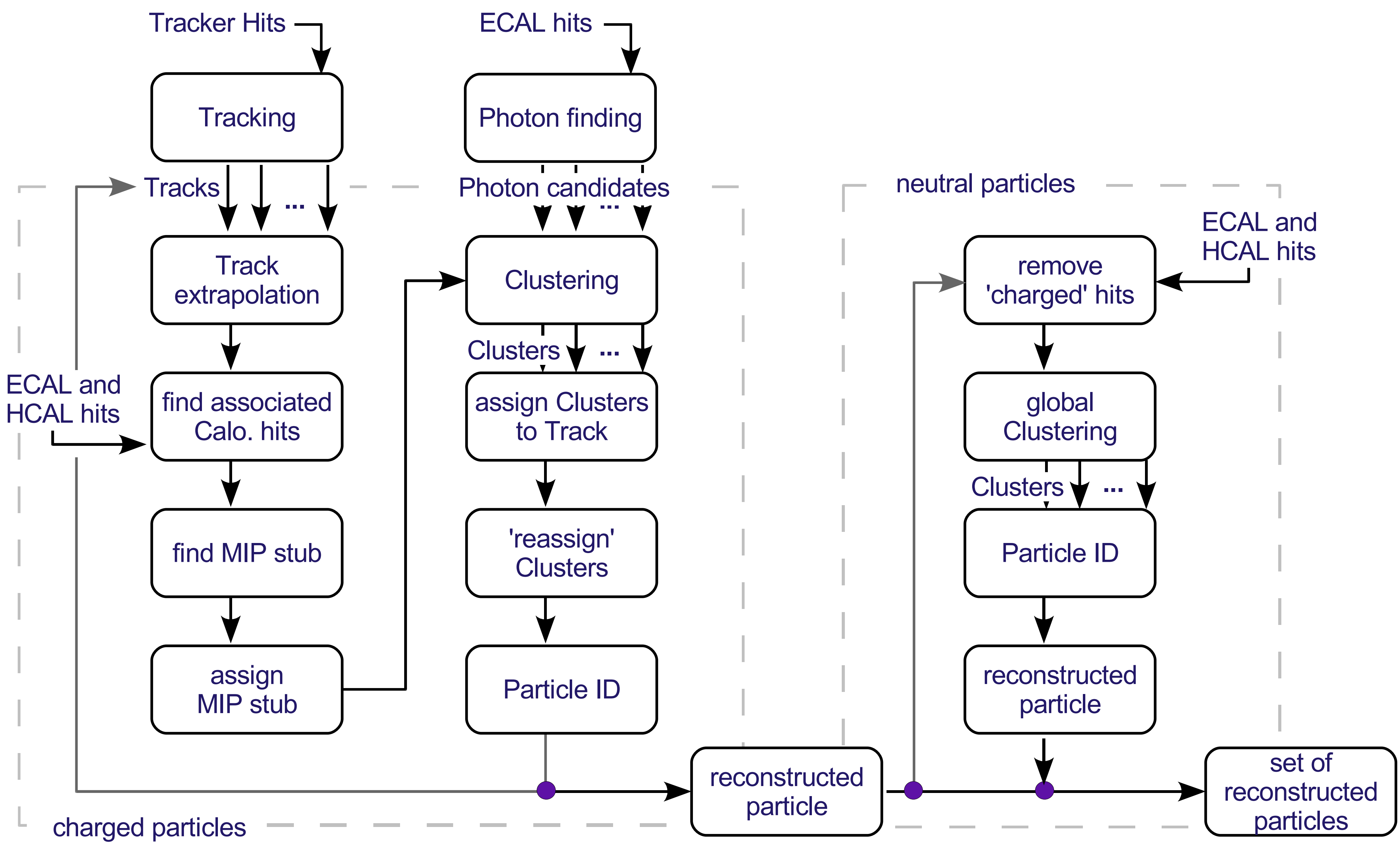
'Philosophy' of TrackBased PFlow:

- tracking system offers the **most accurate** measurement in detector
- → start from tracks, use as much track information as possible (extrapolation, direction, momentum, dE/dx, PID, ...)
- try to establish a **modular** track-based PFlow algorithm in Marlin
- → base on modules and experience made with Wolf (started mid/end 2006)

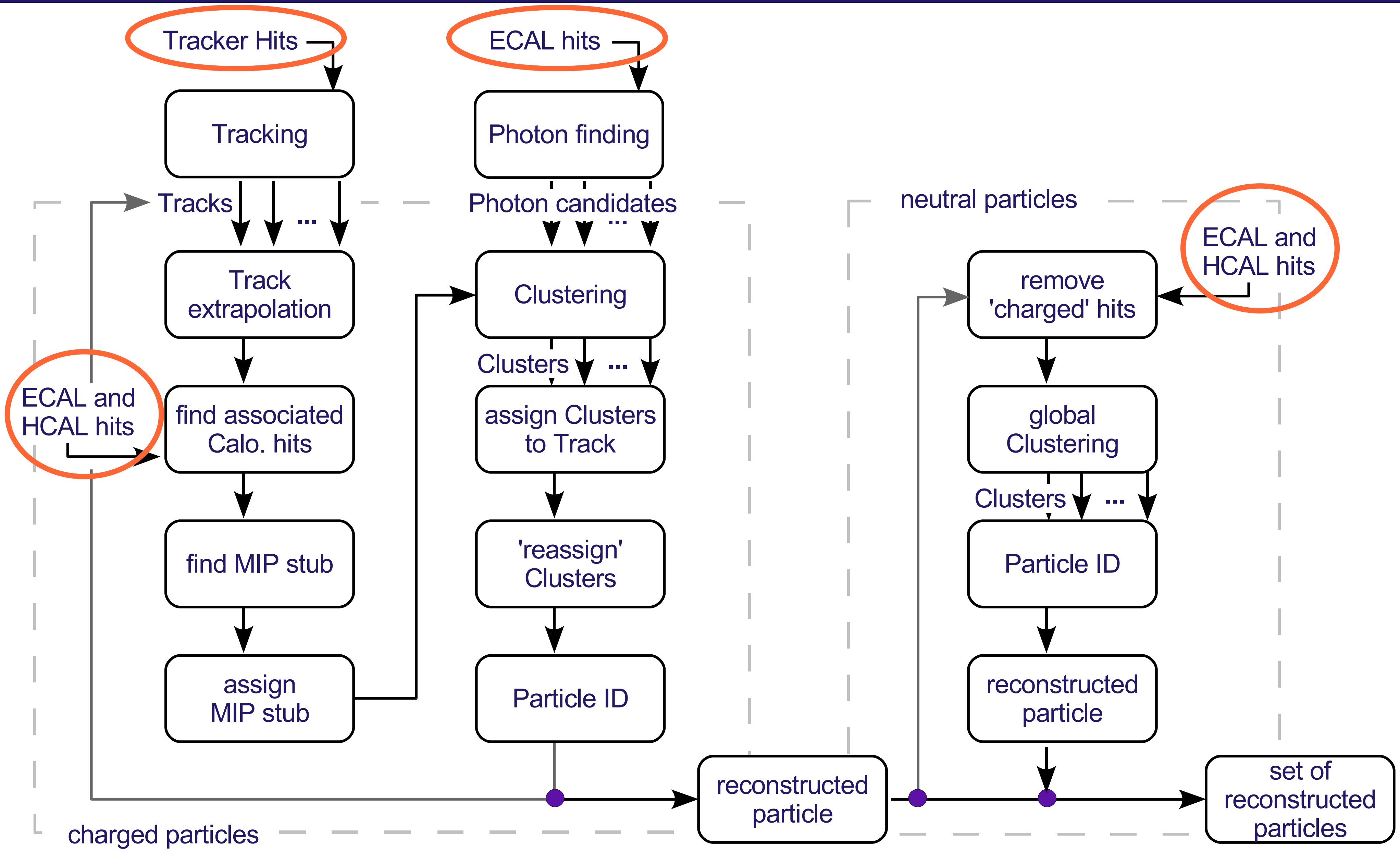




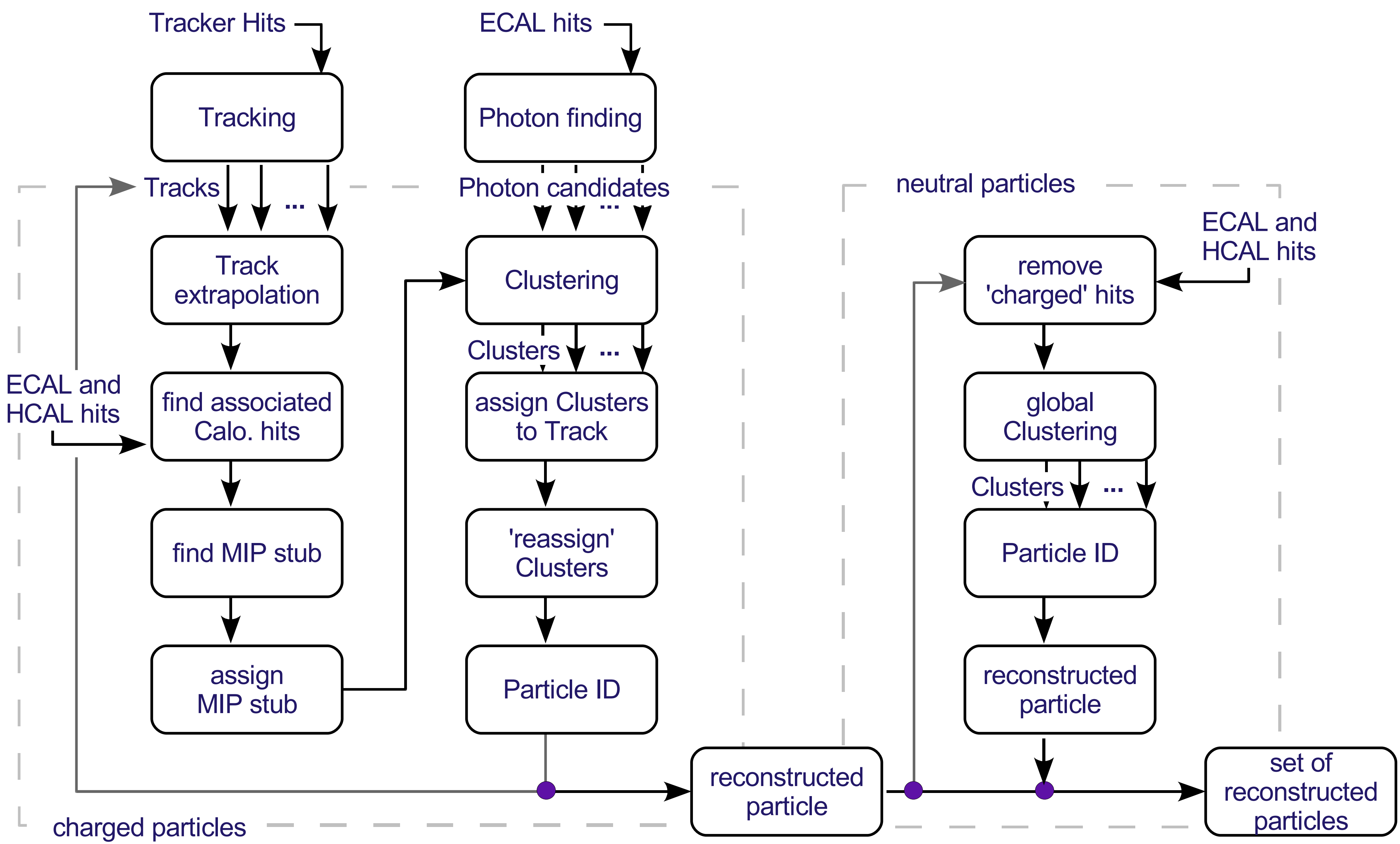
Track-Based Particle Flow Algorithm



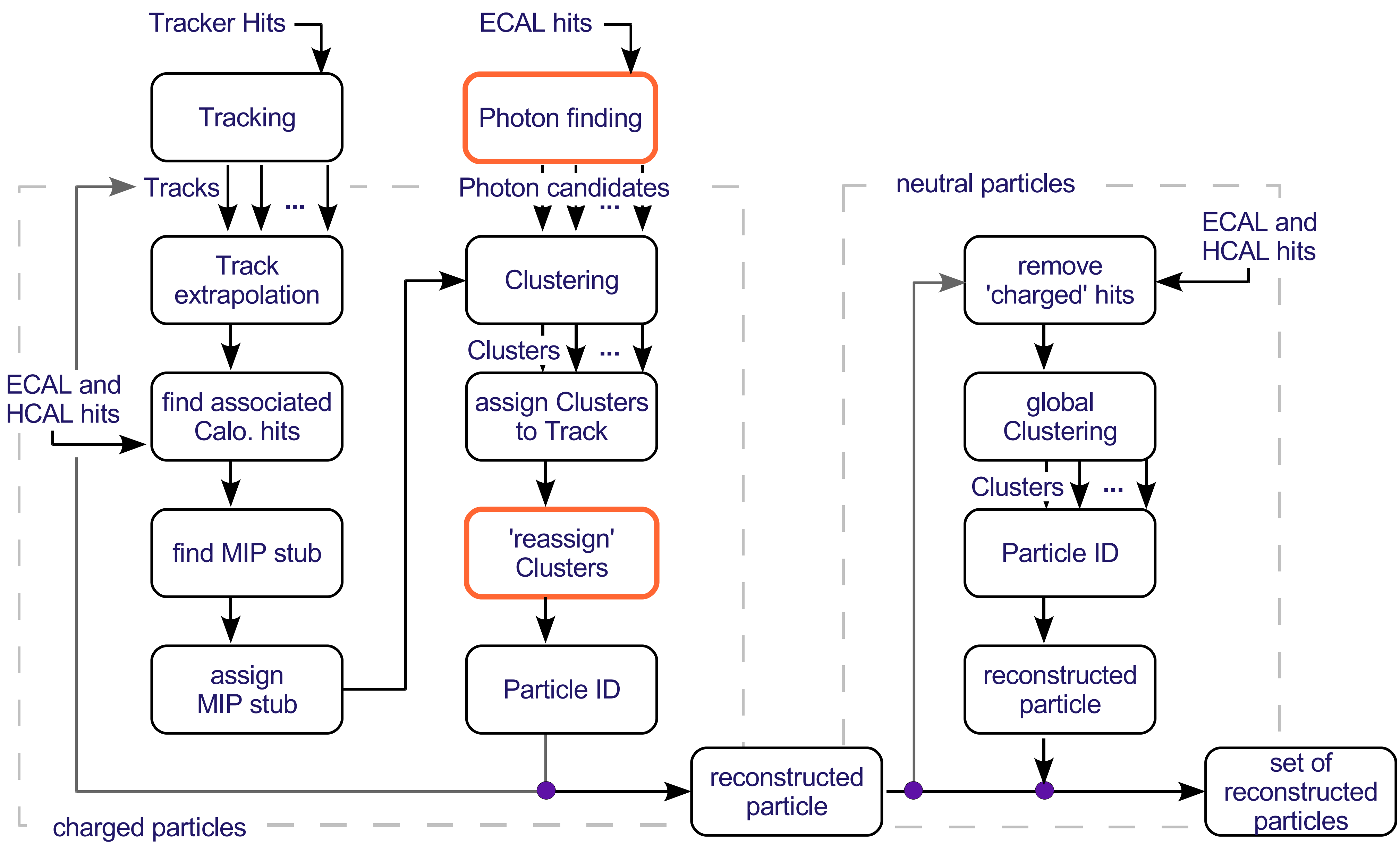
Track-Based Particle Flow Algorithm



Track-Based Particle Flow Algorithm

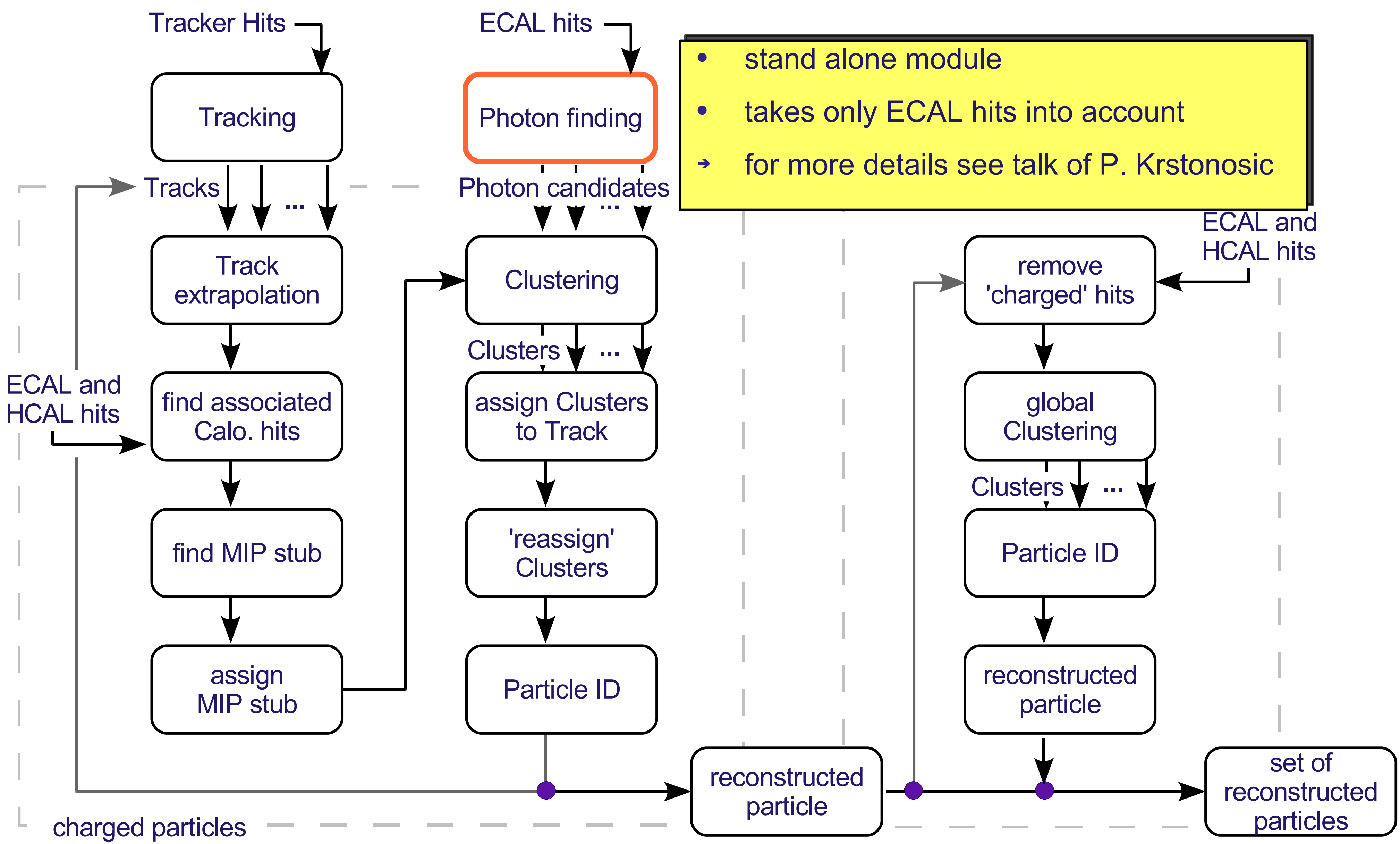


New Modules

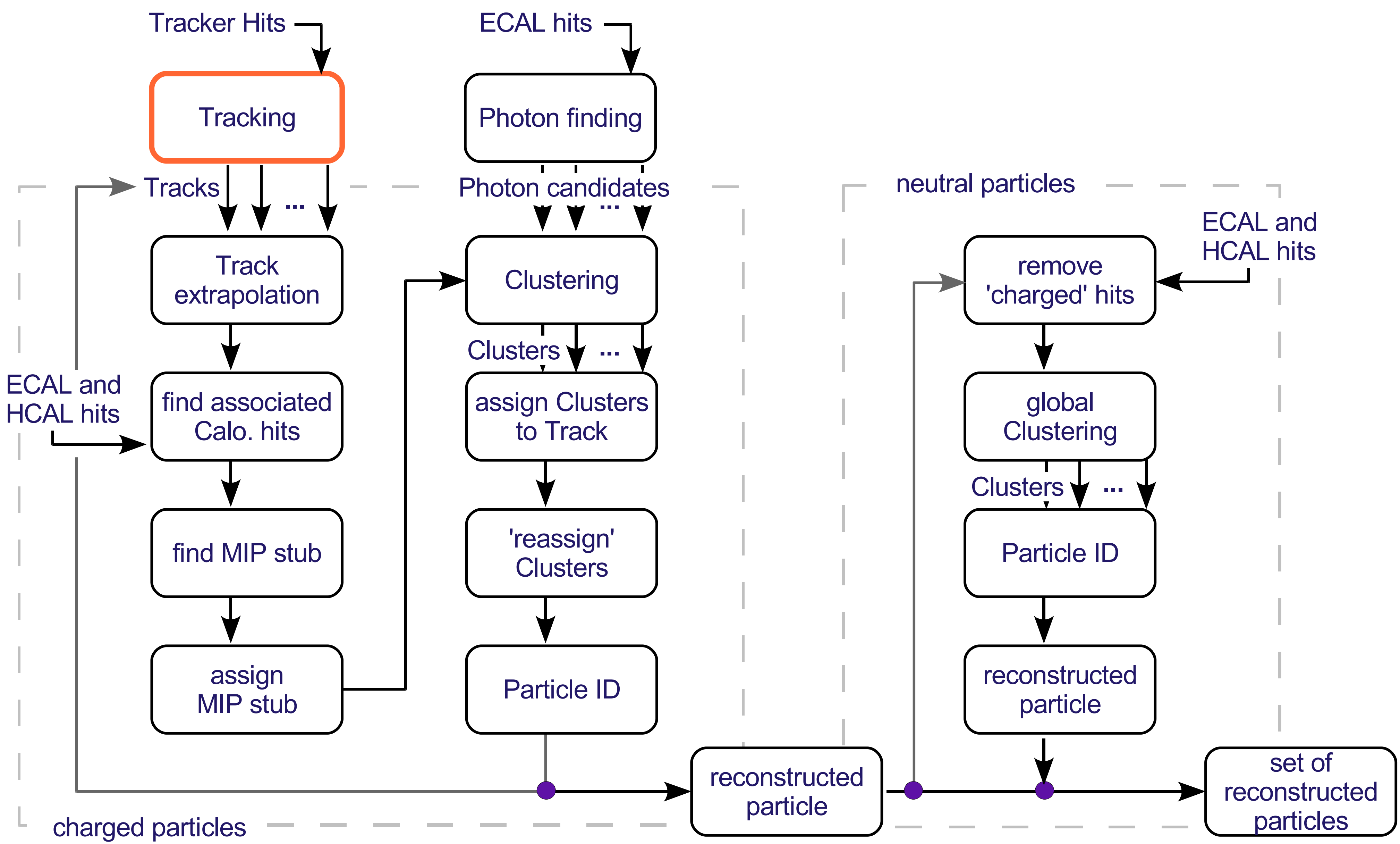

Tracker Hits
ECAL hits
Tracking
Photon finding
Tracks
...
Photon candidates
...
neutral particles
ECAL and HCAL hits
Track extrapolation
Clustering
remove 'charged' hits
Clusters
...
ECAL and HCAL hits
find associated Calo. hits
assign Clusters to Track
global Clustering
Clusters
...
find MIP stub
'reassign' Clusters
Particle ID
assign MIP stub
Particle ID
reconstructed particle
reconstructed particle
set of reconstructed particles
charged particles

Photon Finding

- stand alone module
- takes only ECAL hits into account
- → for more details see talk of P. Krstonosic



Tracking

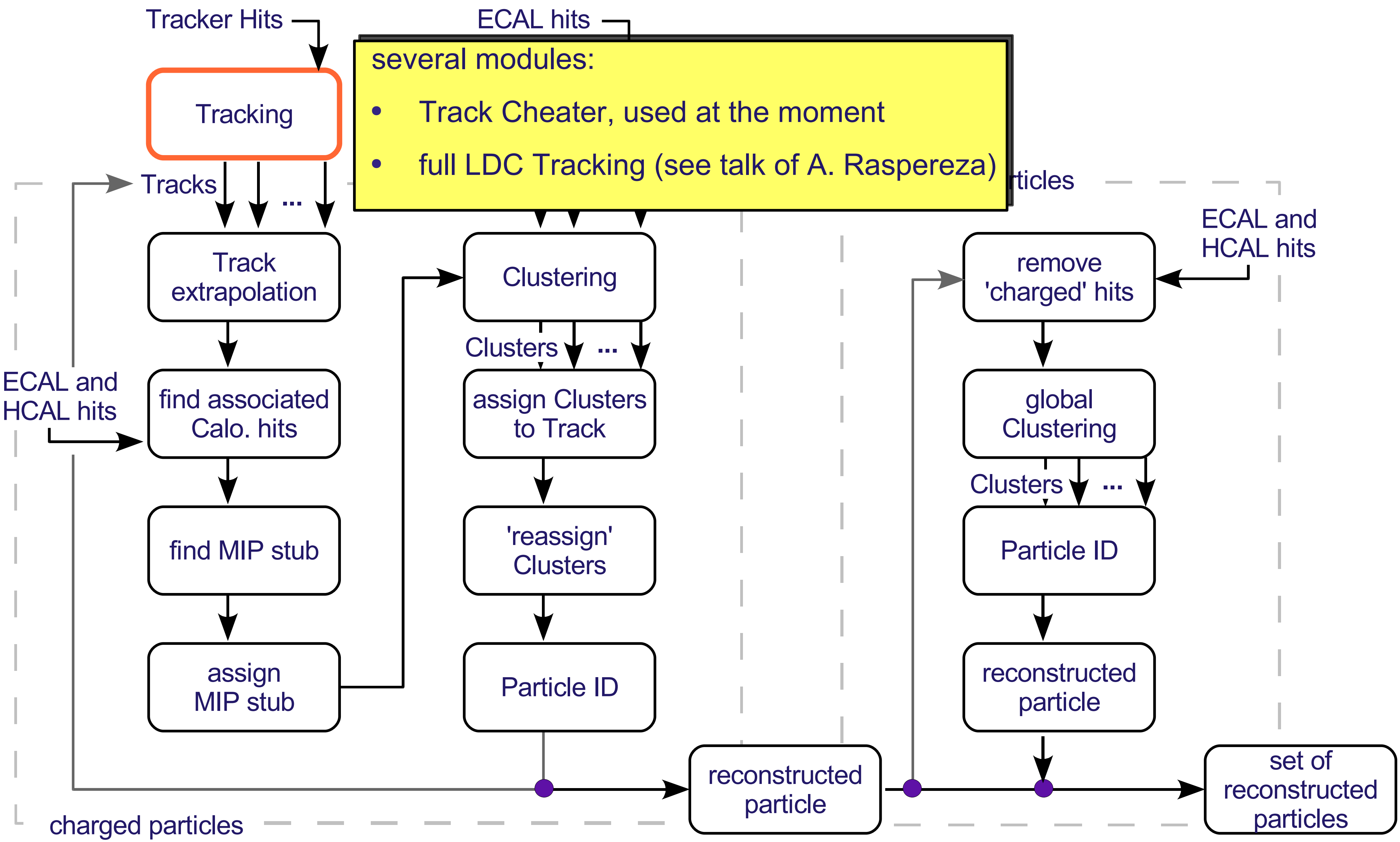


Tracker Hits
ECAL hits
Tracking
Photon finding
Tracks
...
Photon candidates
...
neutral particles
ECAL and
HCAL hits
Track
extrapolation
Clustering
remove
'charged' hits
Clusters
...
ECAL and
HCAL hits
find associated
Calo. hits
assign Clusters
to Track
global
Clustering
Clusters
...
find MIP stub
'reassign'
Clusters
Particle ID
assign
MIP stub
Particle ID
reconstructed
particle
reconstructed
particle
set of
reconstructed
particles
charged particles

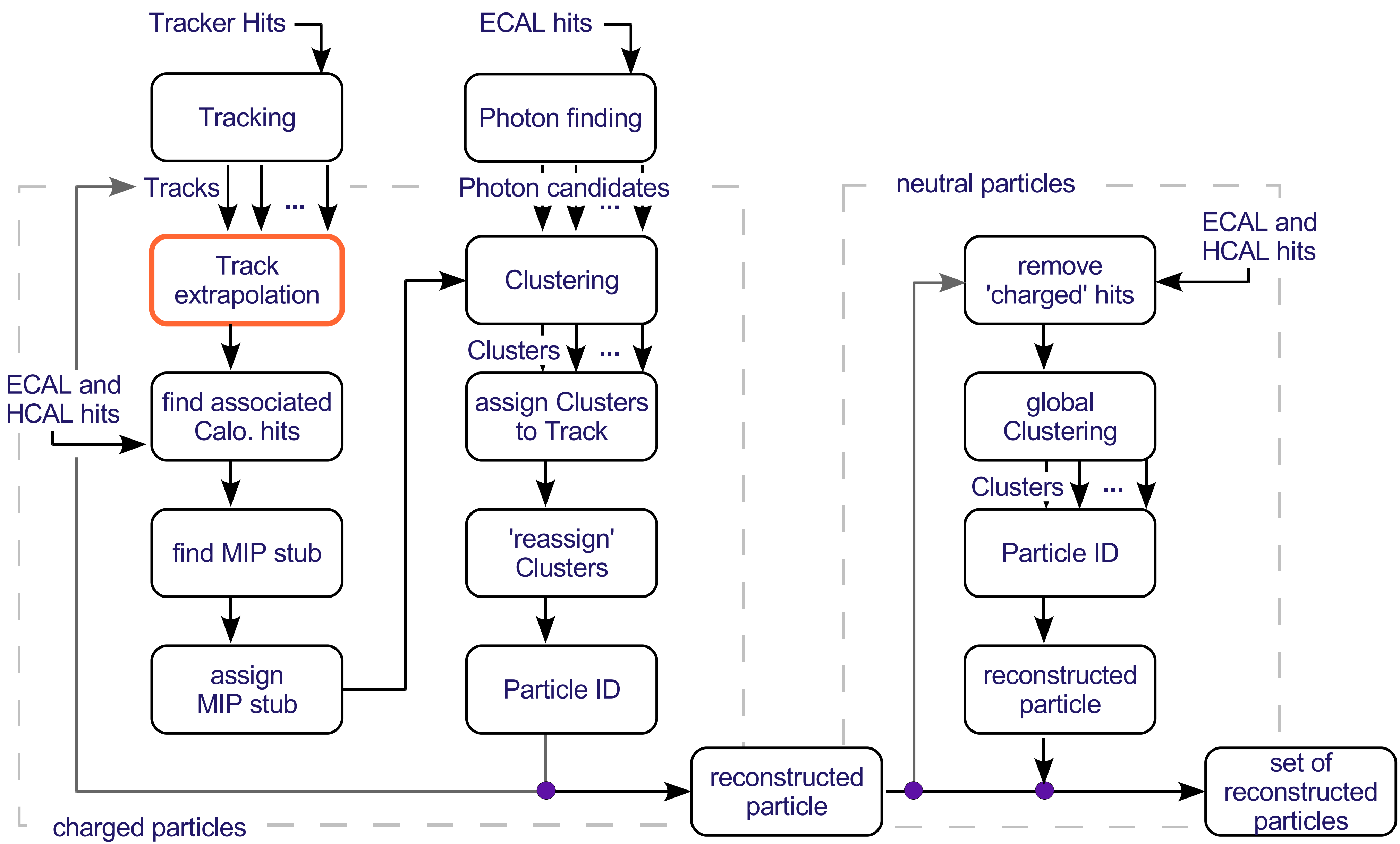
Tracking

several modules:

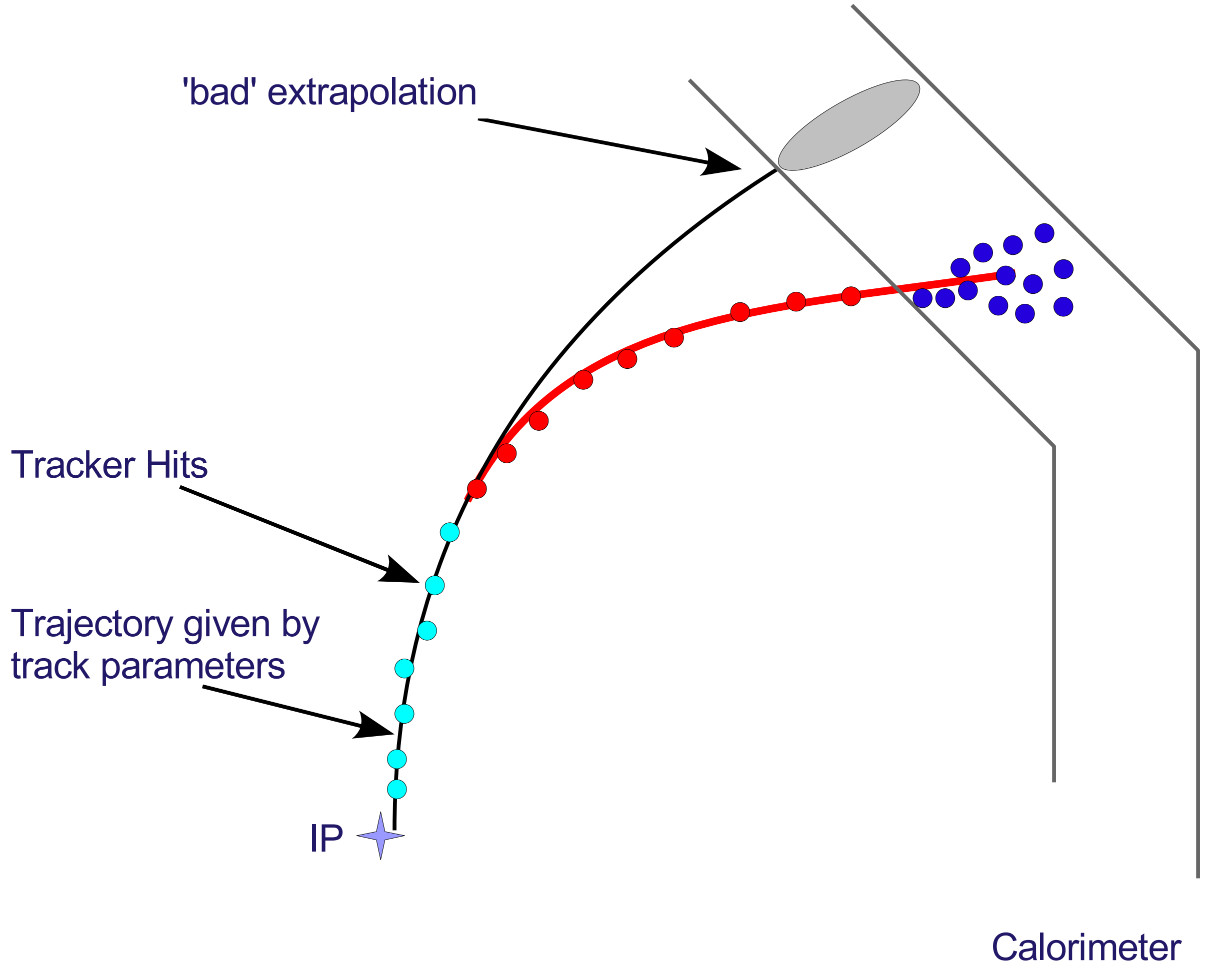
- Track Cheater, used at the moment
- full LDC Tracking (see talk of A. Raspereza)



Track Extrapolation



Track Extrapolation



- Track parameters might give a 'bad' extrapolation into the Calorimeter

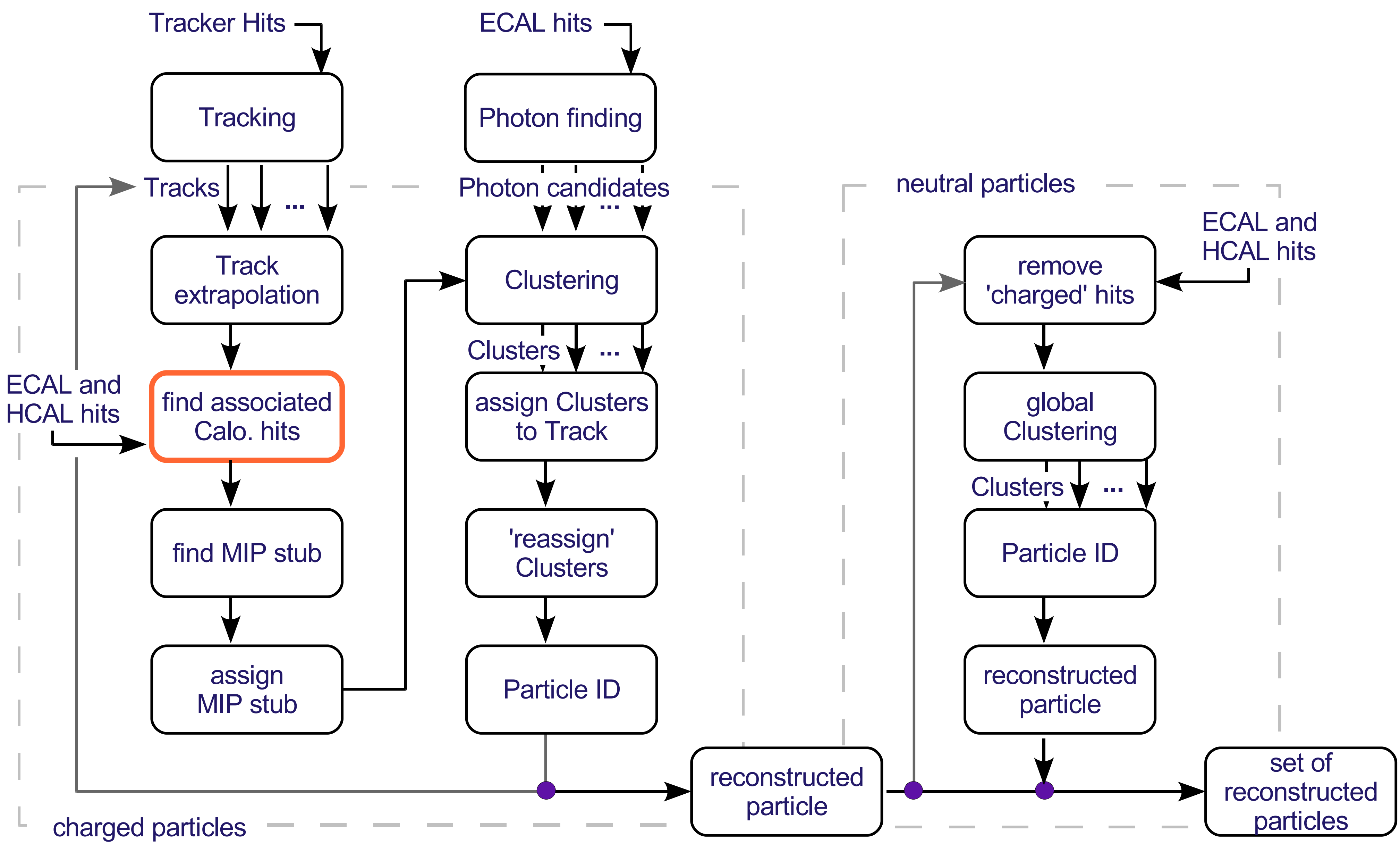
<u>Algorithm:</u>

- take the outermost n Tracker hits (w.r.t path-length)
- fit trajectory on these hits
- ➔ at the moment done with a simple helix model (w/o energy loss, multiple scattering, ...)

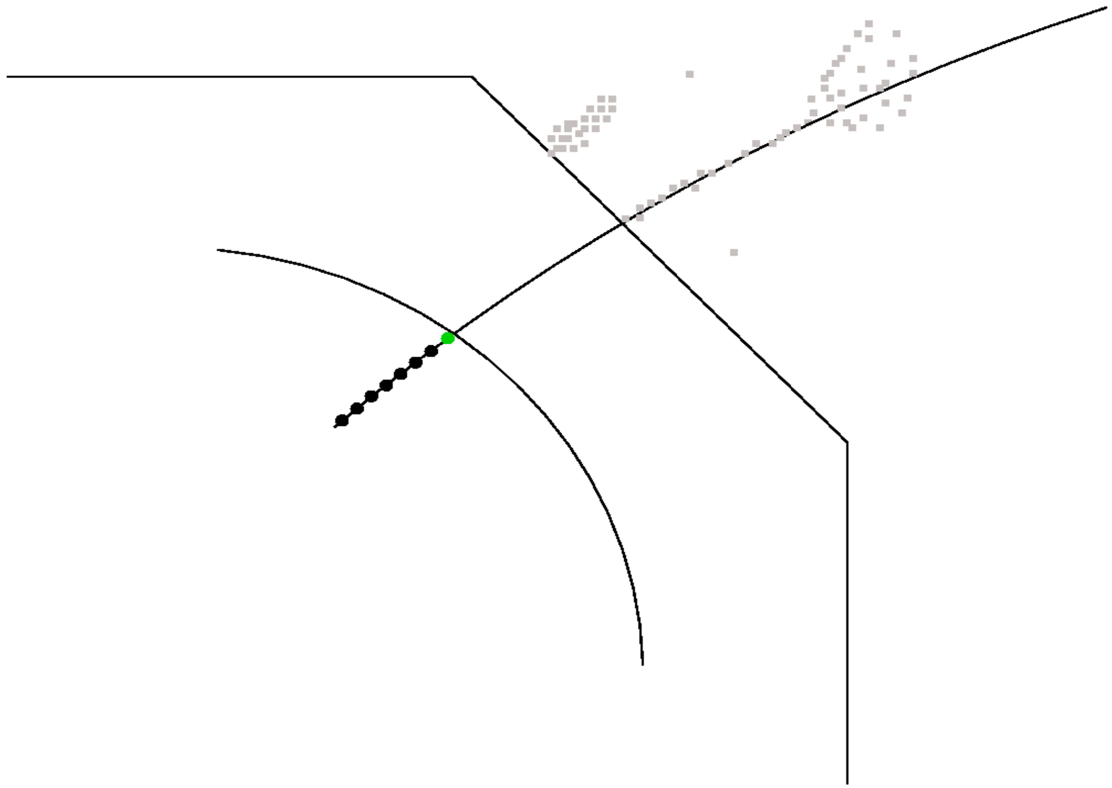
<u>Technicalities:</u>

- based on the Trajectory interface in MarlinUtil
- fitting based on GSL using canonical track parametrisation
- ➔ more general fitting module for different Trajectory 'models' needed

Find associated Calorimeter Hits



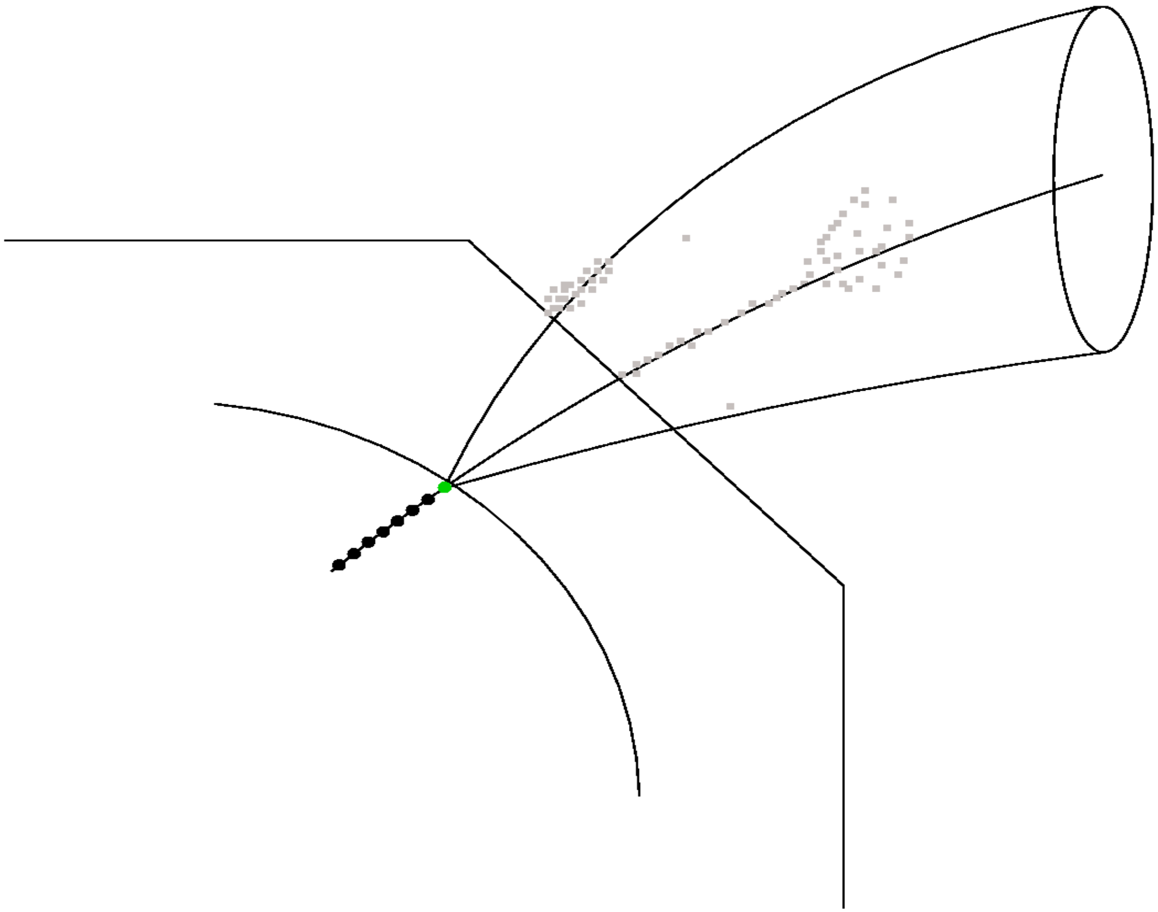
Find associated Calorimeter Hits



Find associated Calorimeter Hits

<u>Algorithm:</u>

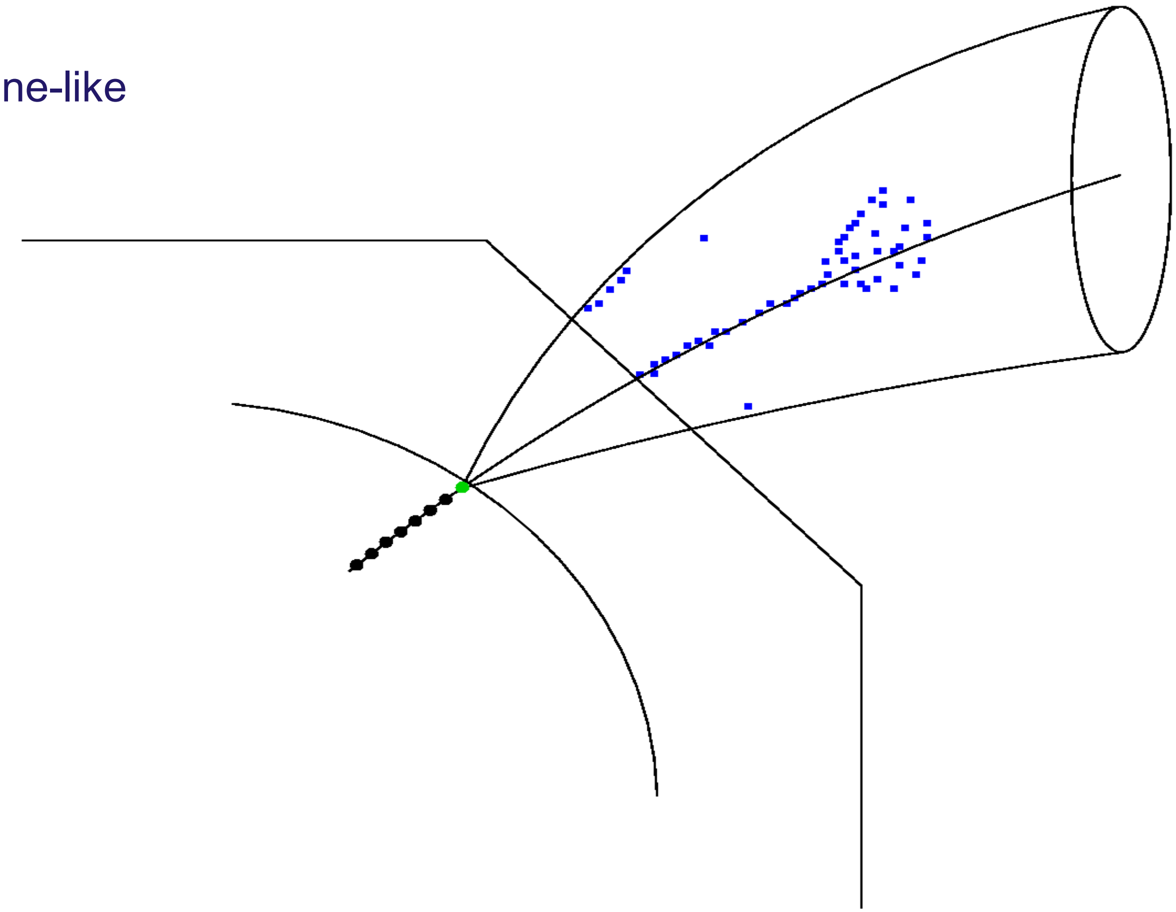
- put cone-like tube around extrapolated trajectory



Find associated Calorimeter Hits

Algorithm:

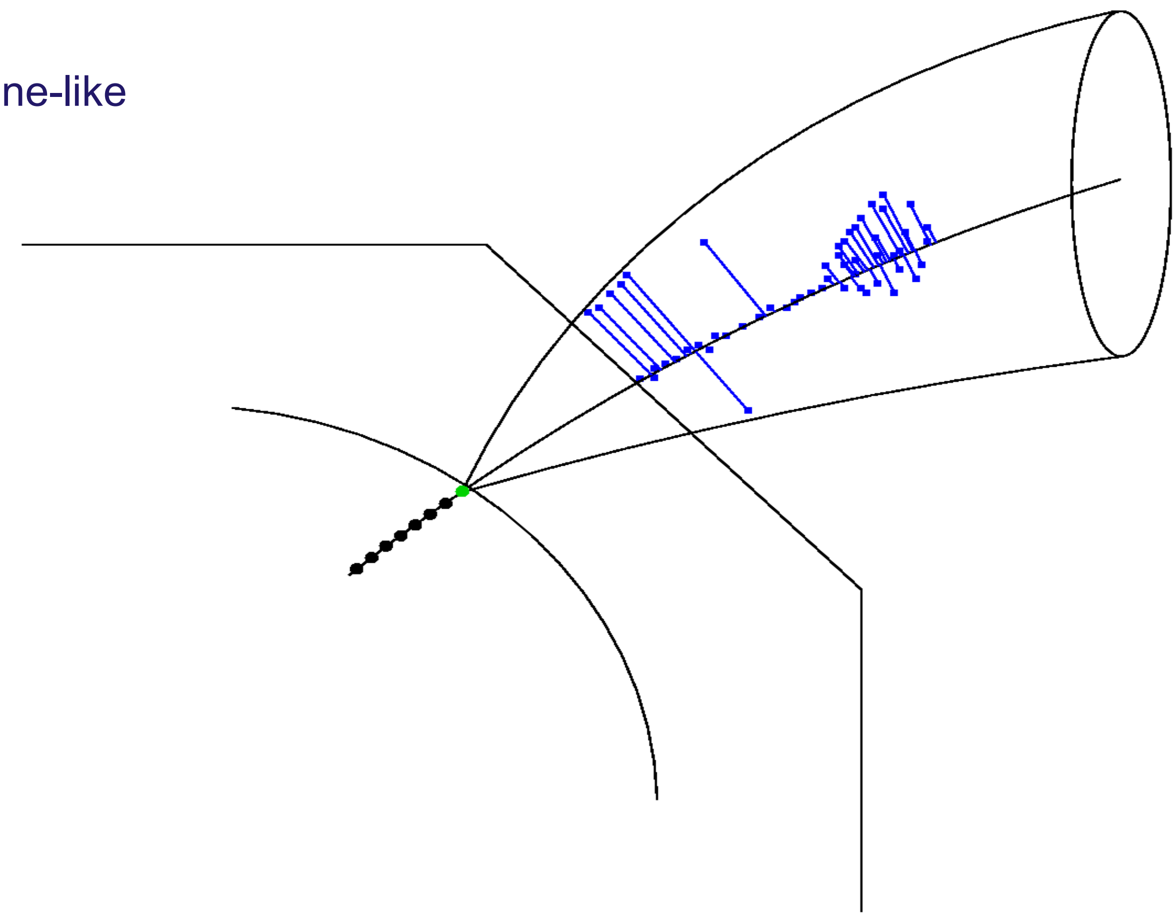
- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube



Find associated Calorimeter Hits

Algorithm:

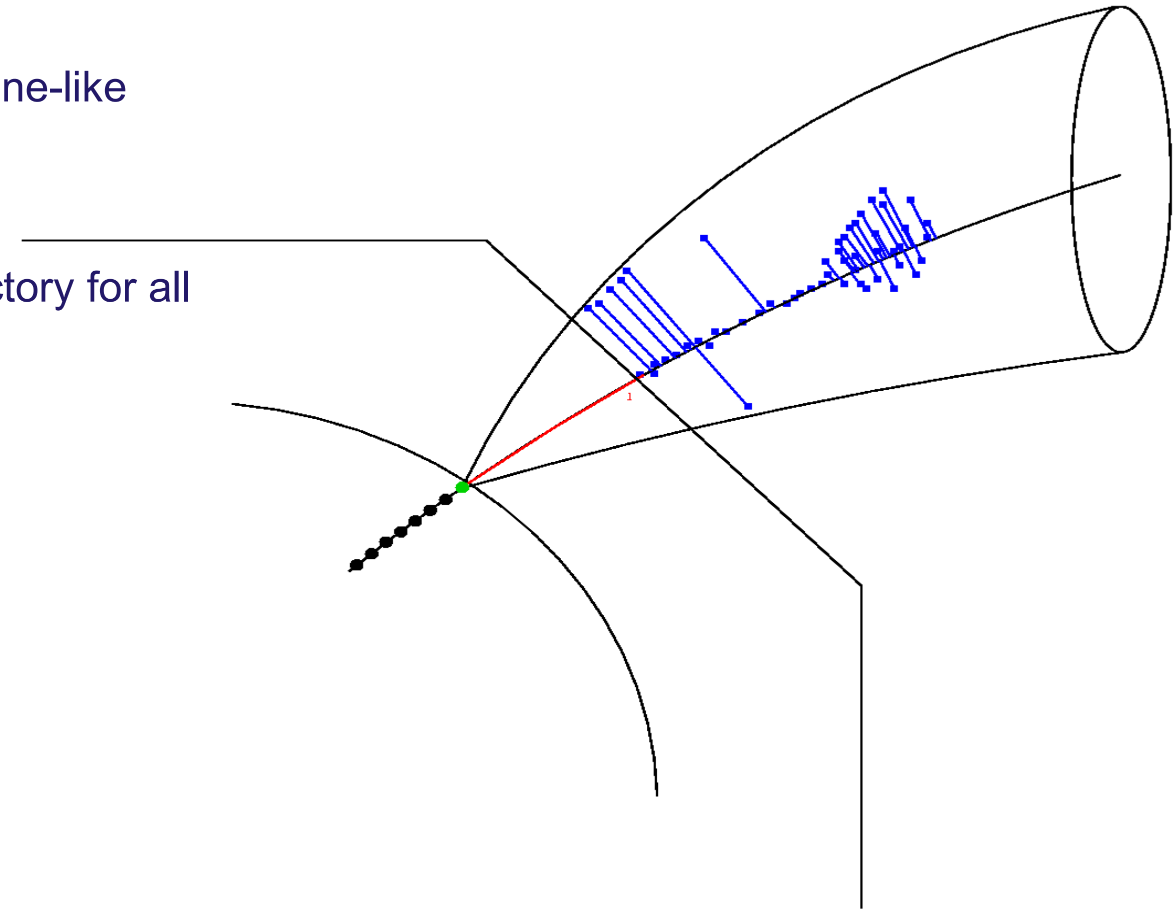
- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube
- project all hits on trajectory



Find associated Calorimeter Hits

Algorithm:

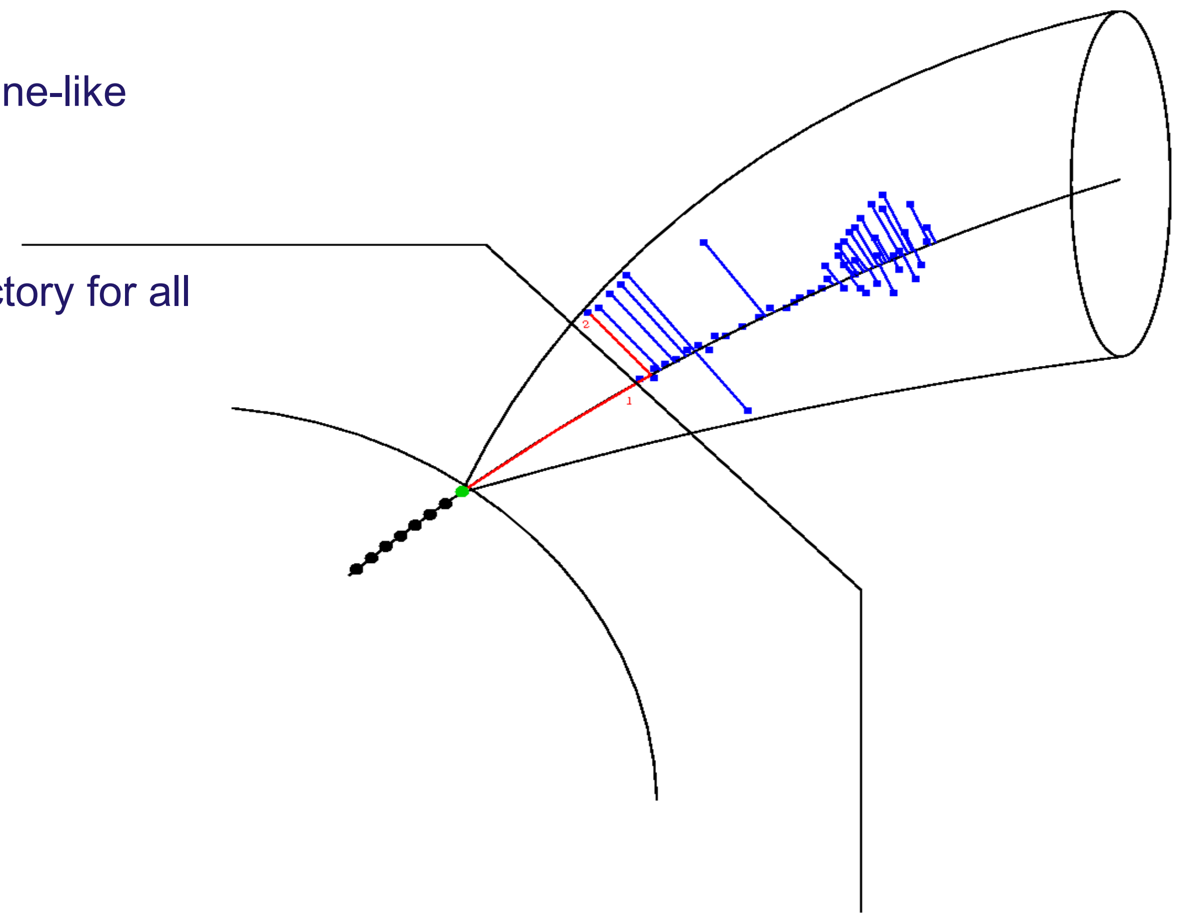
- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube
- project all hits on trajectory
- calculate path length on trajectory for all hits



Find associated Calorimeter Hits

Algorithm:

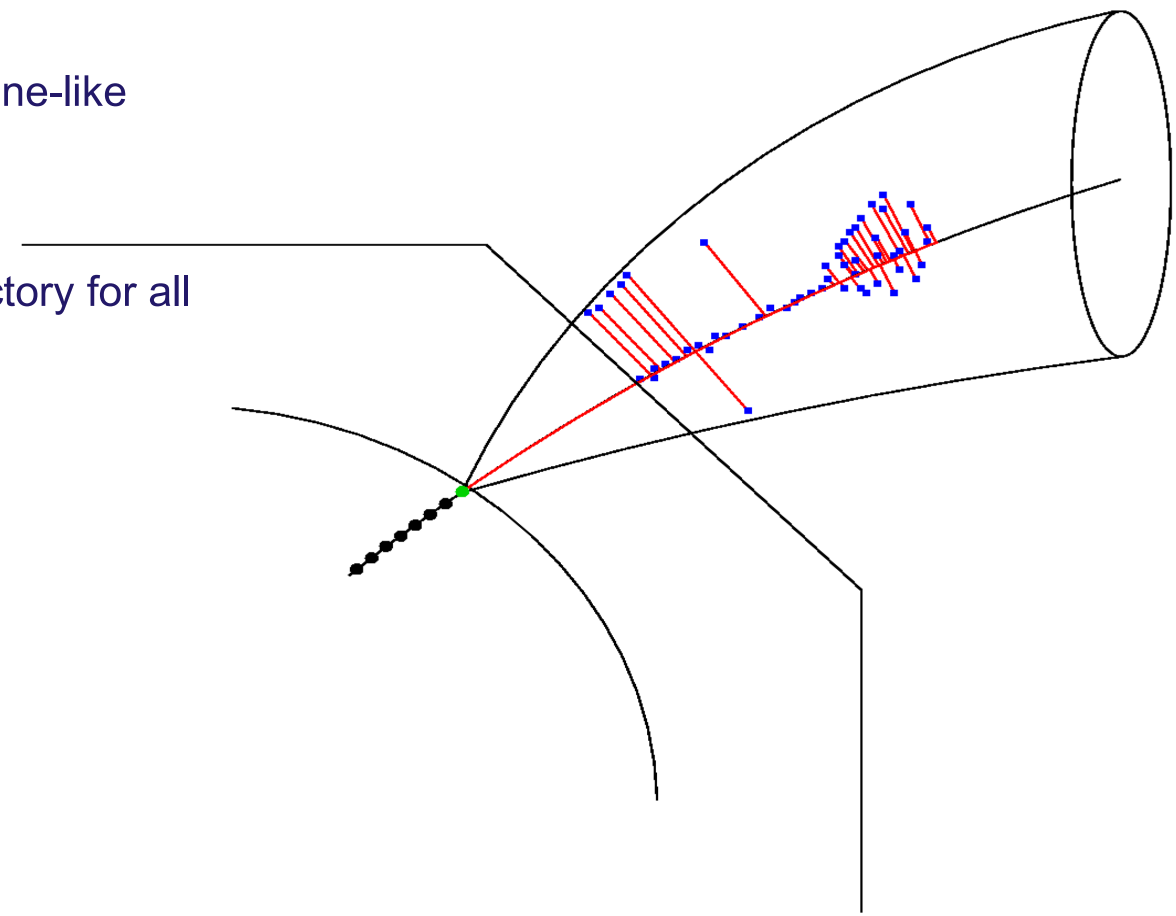
- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube
- project all hits on trajectory
- calculate path length on trajectory for all hits



Find associated Calorimeter Hits

<u>Algorithm:</u>

- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube
- project all hits on trajectory
- calculate path length on trajectory for all hits
- sort hits by their path lengths

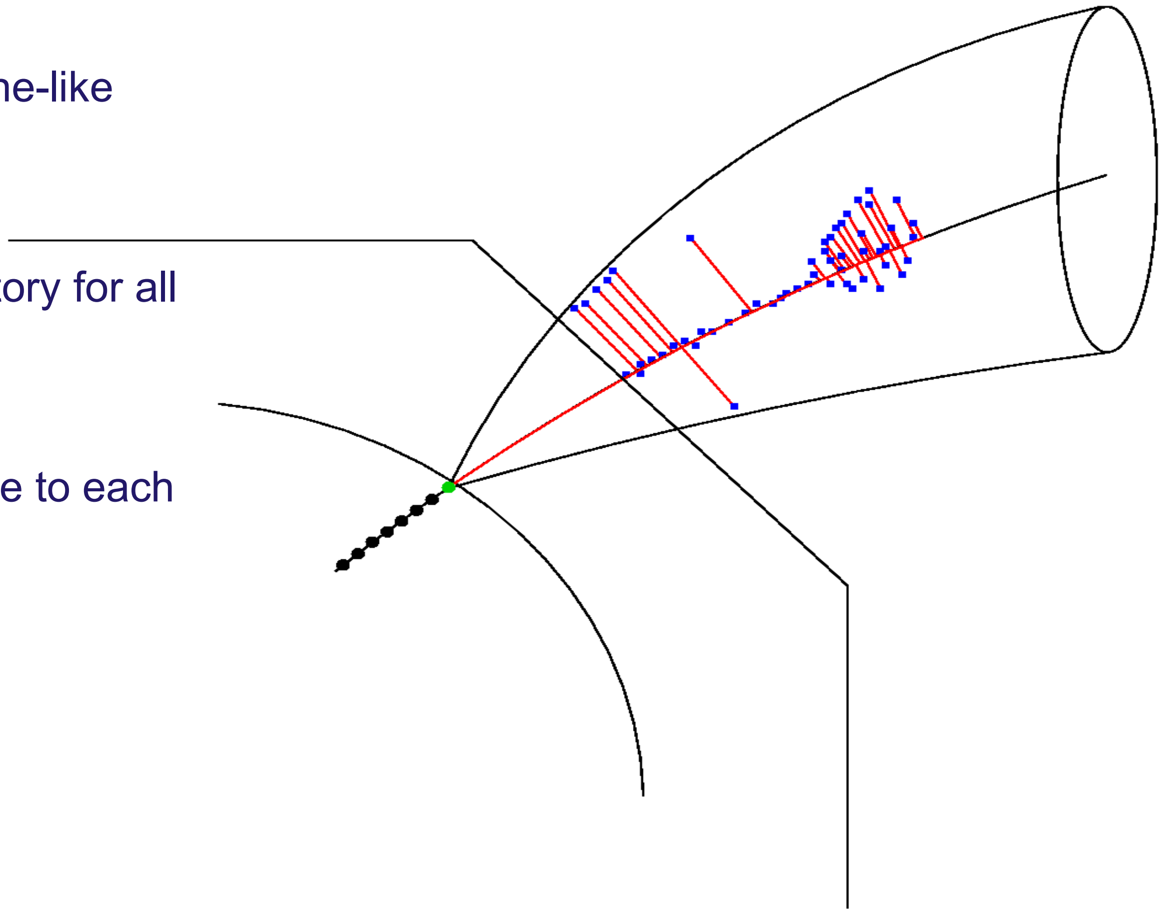


Find associated Calorimeter Hits

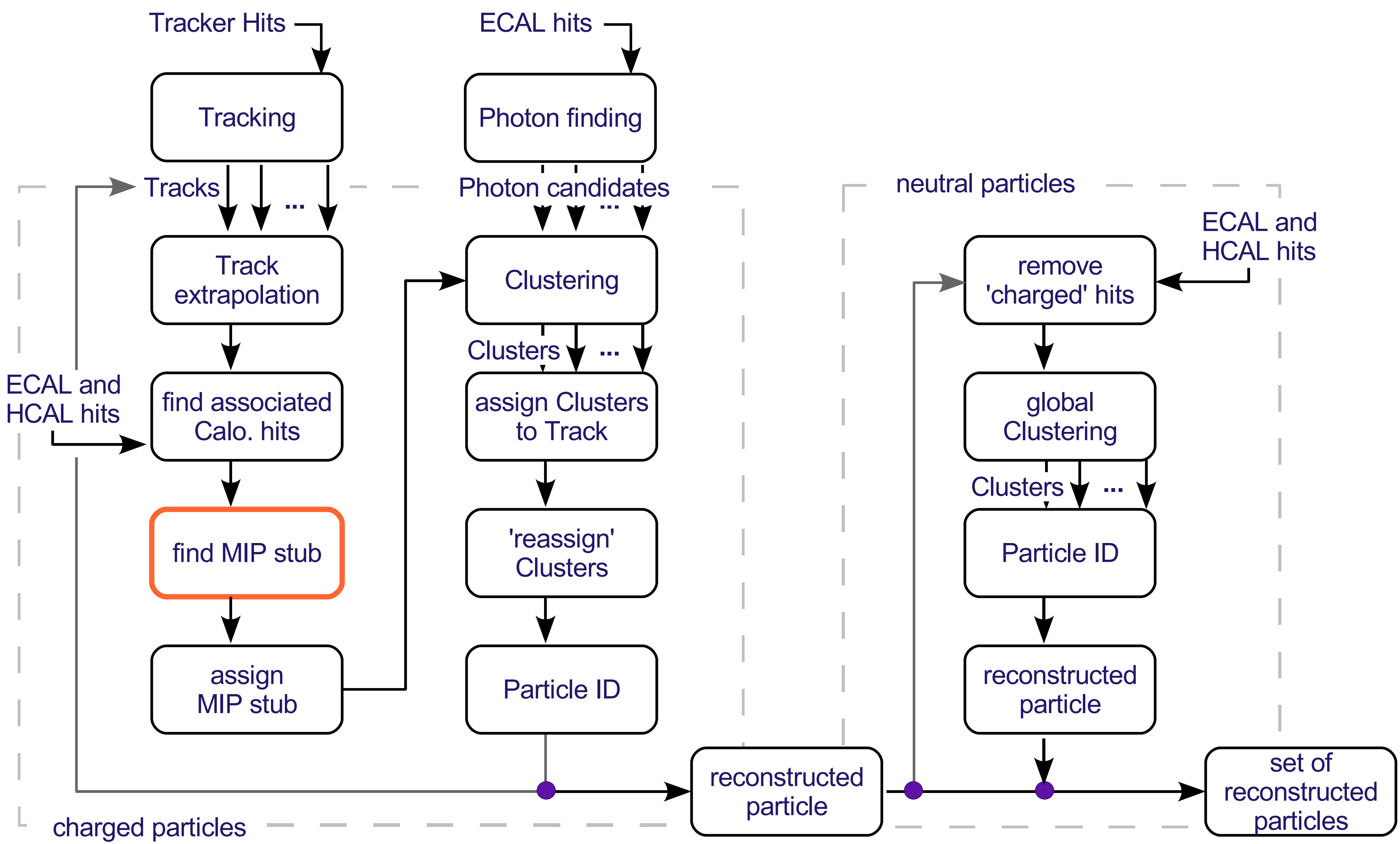
Algorithm:

- put cone-like tube around extrapolated trajectory
- cut calorimeter hits outside cone-like tube
- project all hits on trajectory
- calculate path length on trajectory for all hits
- sort hits by their path lengths

➔ assign path length and distance to each hit



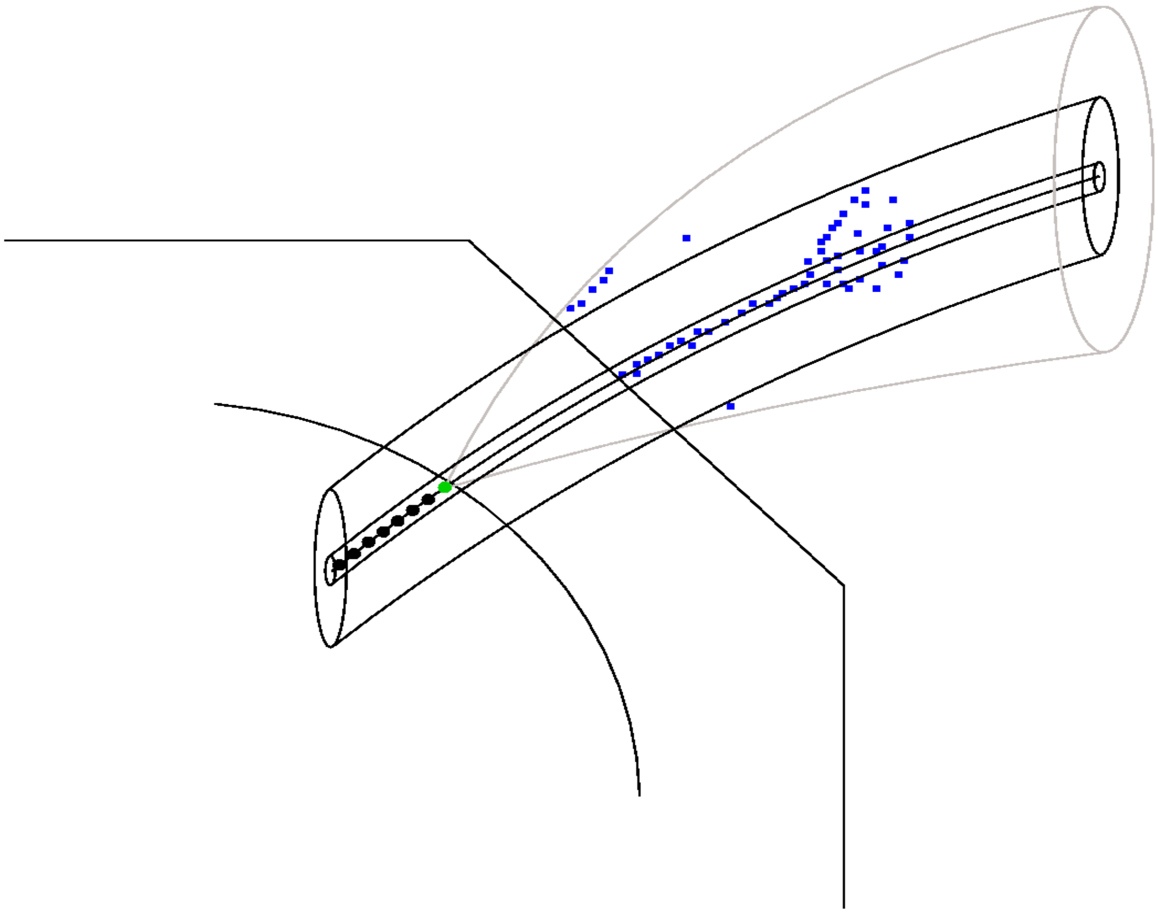
Find MIP Stub



Find MIP Stub

Algorithm:

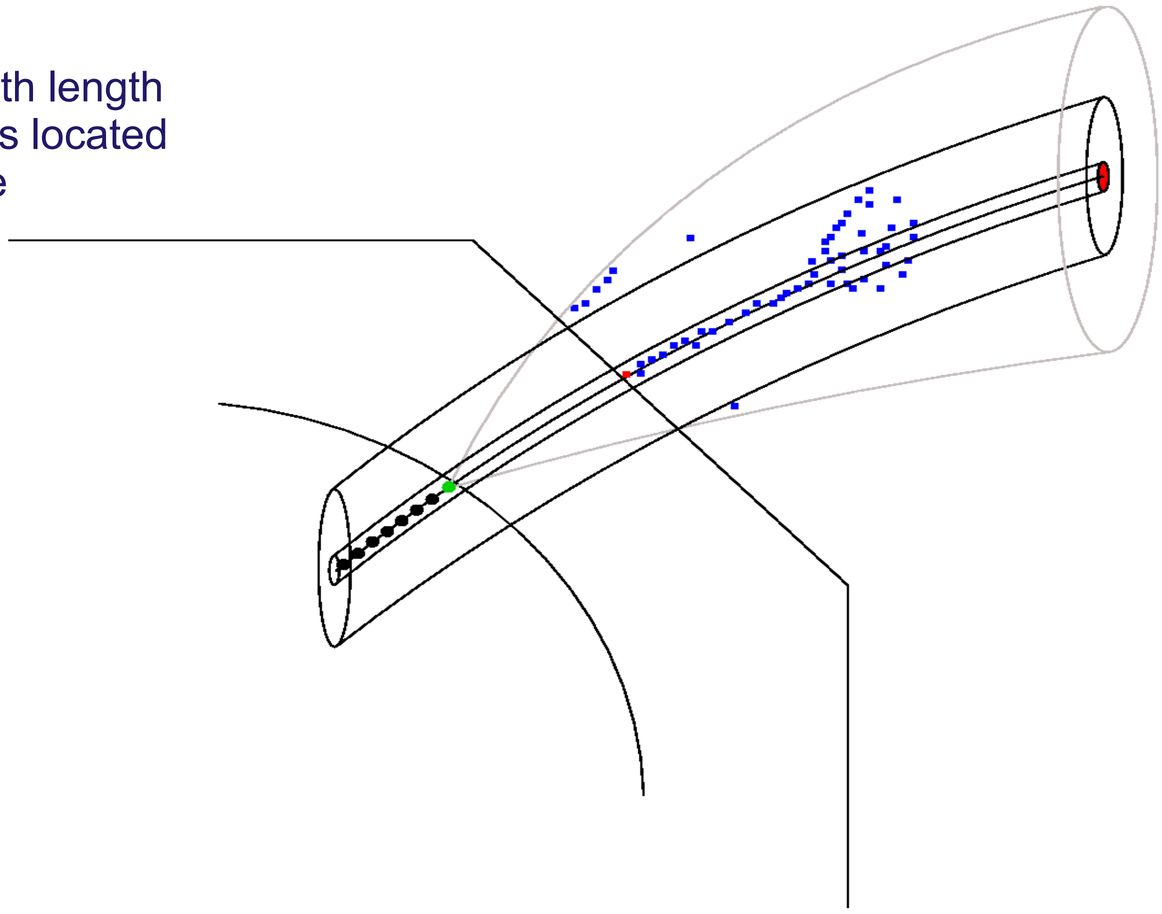
- put two cylindrical tubes around extrapolated trajectory



Find MIP Stub

Algorithm:

- put two cylindrical tubes around extrapolated trajectory
- take first hit according to its path length and add it to the MIP stub if it is located inside the inner cylindrical tube

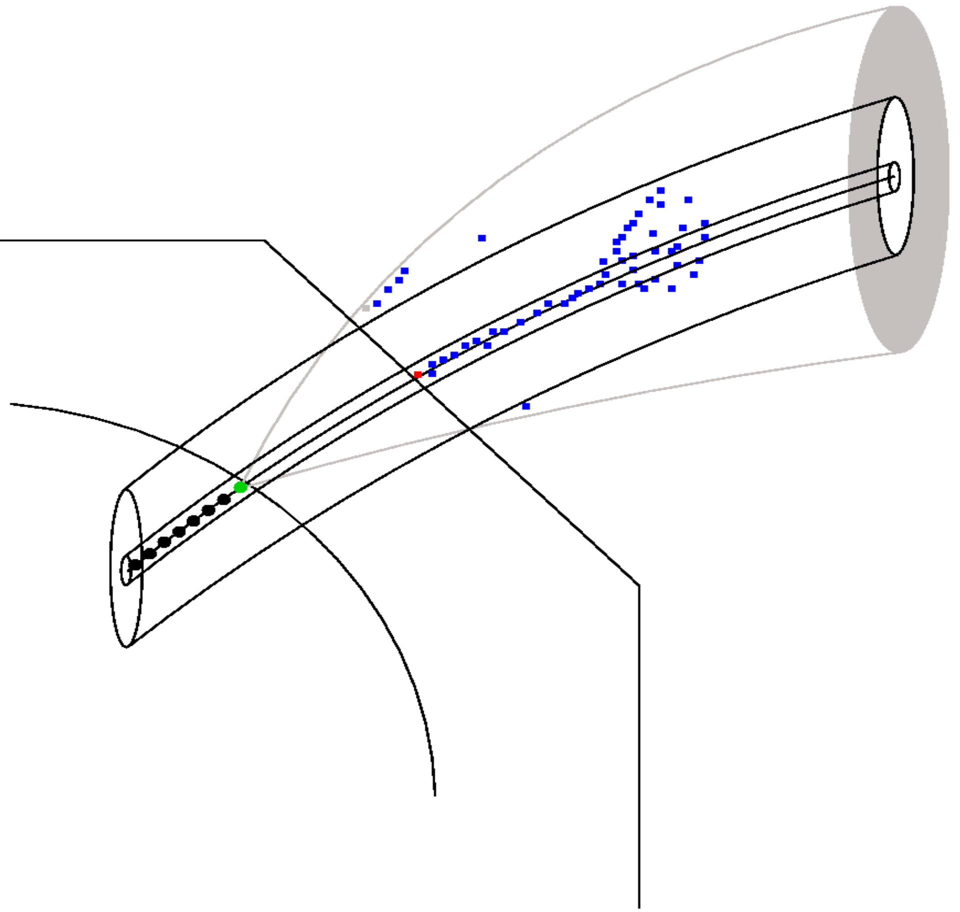


Find MIP Stub

Algorithm:

- put two cylindrical tubes around extrapolated trajectory
- take first hit according to its path length and add it to the MIP stub if it is located inside the inner cylindrical tube
- take the next hit and discard it if it is located outside the outer tube

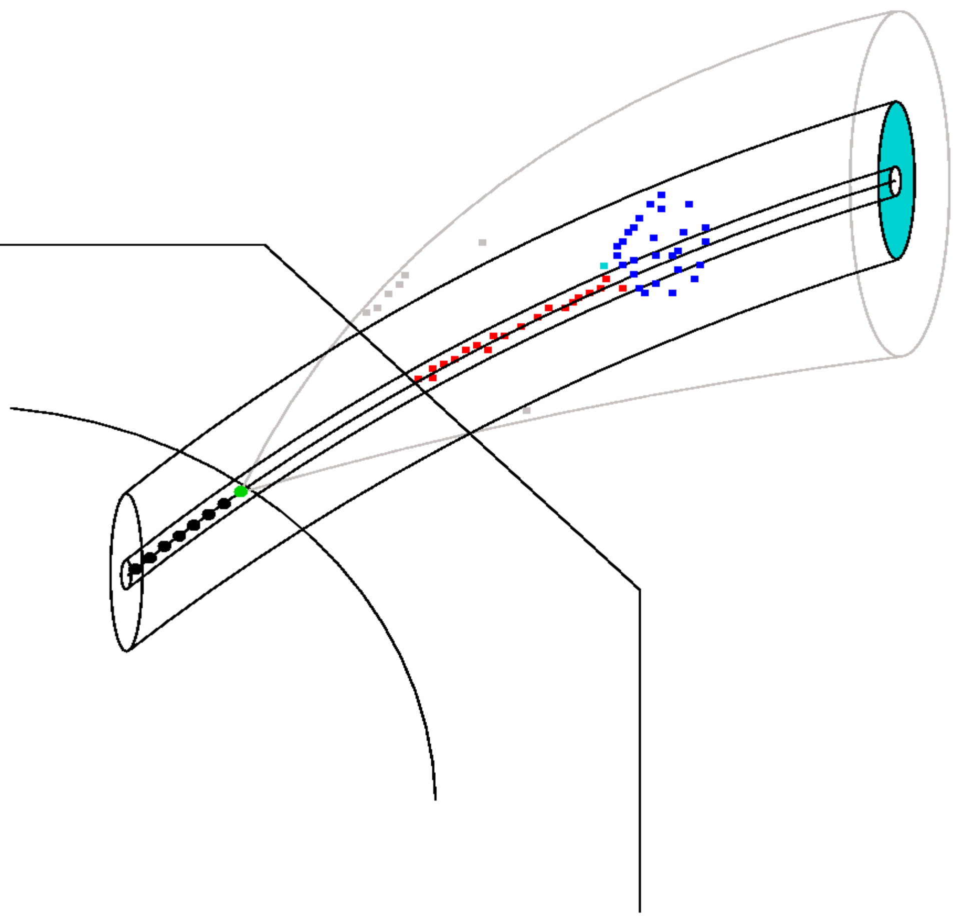
→ amplitude information can be taken into account (MIP like)



Find MIP Stub

Algorithm:

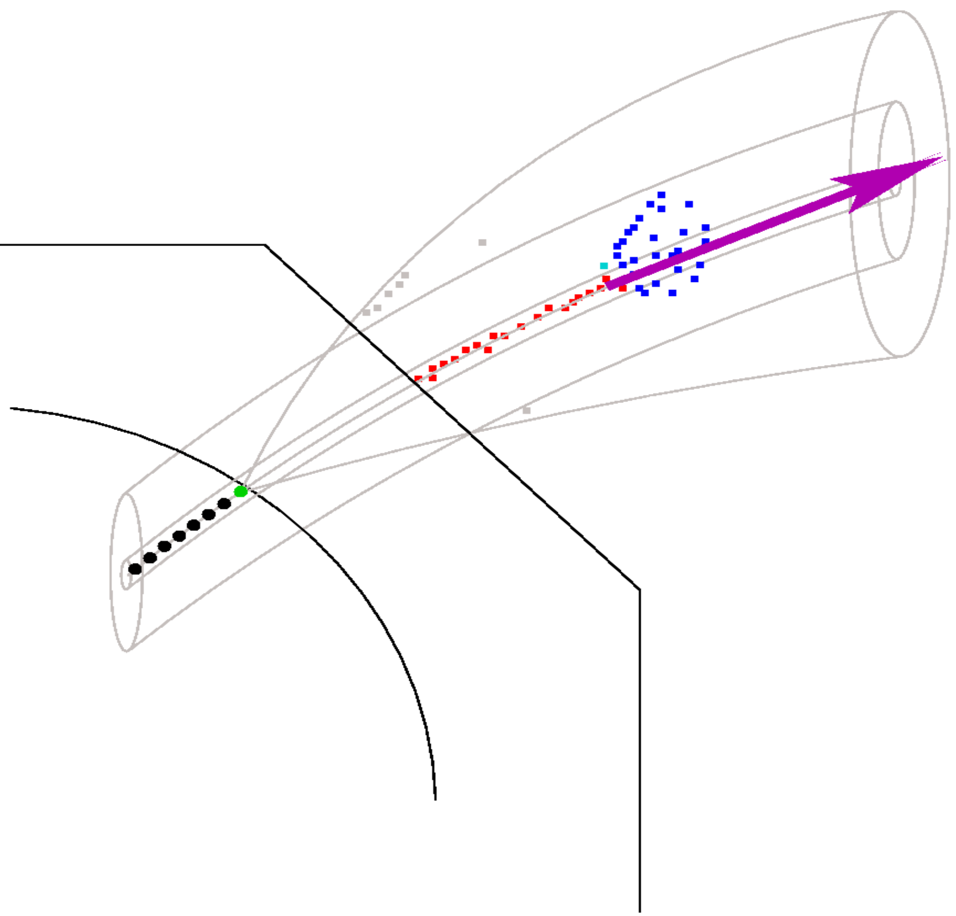
- put two cylindrical tubes around extrapolated trajectory
- take first hit according to its path length and add it to the MIP stub if it is located inside the inner cylindrical tube
- take the next hit and discard it if it is located outside the outer tube
- → amplitude information can be taken into account (MIP like)
- repeat this procedure for all hits until a hit outside the inner and inside the outer cylinder tube is found ('veto-cylinder')



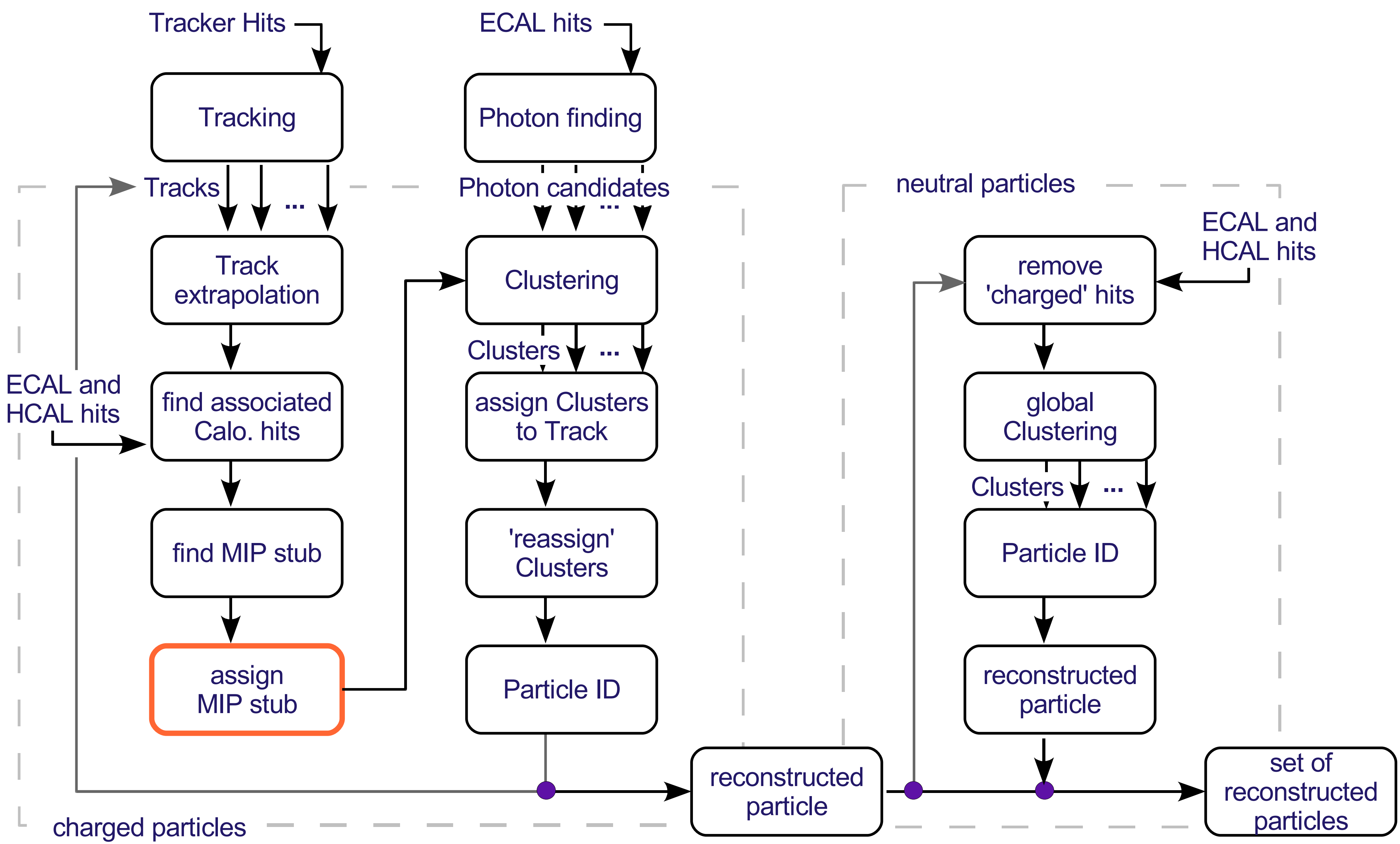
Find MIP Stub

<u>Algorithm:</u>

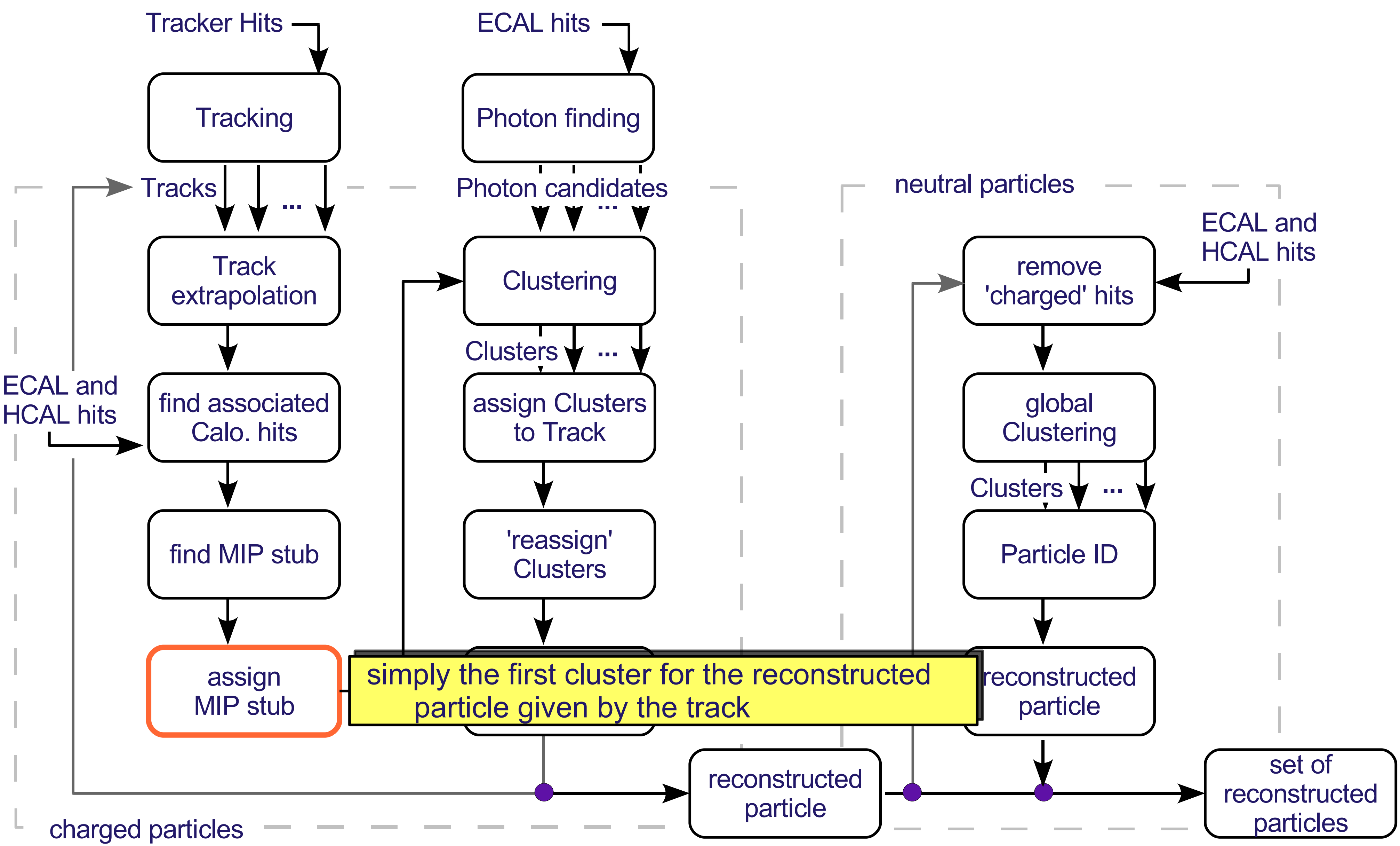
- put two cylindrical tubes around extrapolated trajectory
- take first hit according to its path length and add it to the MIP stub if it is located inside the inner cylindrical tube
- take the next hit and discard it if it is located outside the outer tube
- ➔ amplitude information can be taken into account (MIP like)
- repeat this procedure for all hits until a hit outside the inner and inside the outer cylinder tube is found ('veto-cylinder')
- ➔ stop the MIP stub finding
- take the projection of the last hit collected for the MIP stub as a start point for clustering
- take the direction (tangent) of this point as a start direction for clustering



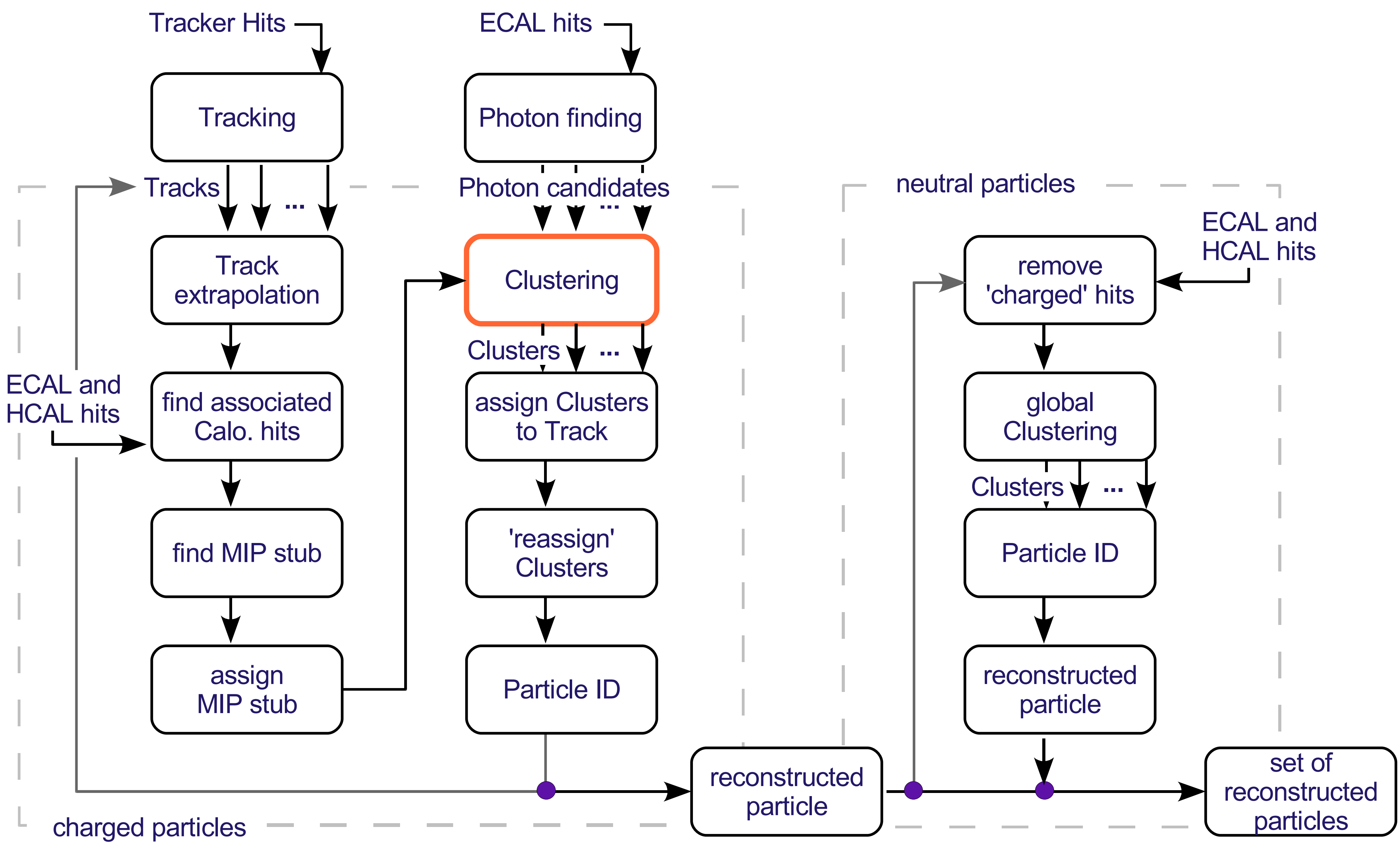
Assign MIP Stub



Assign MIP Stub

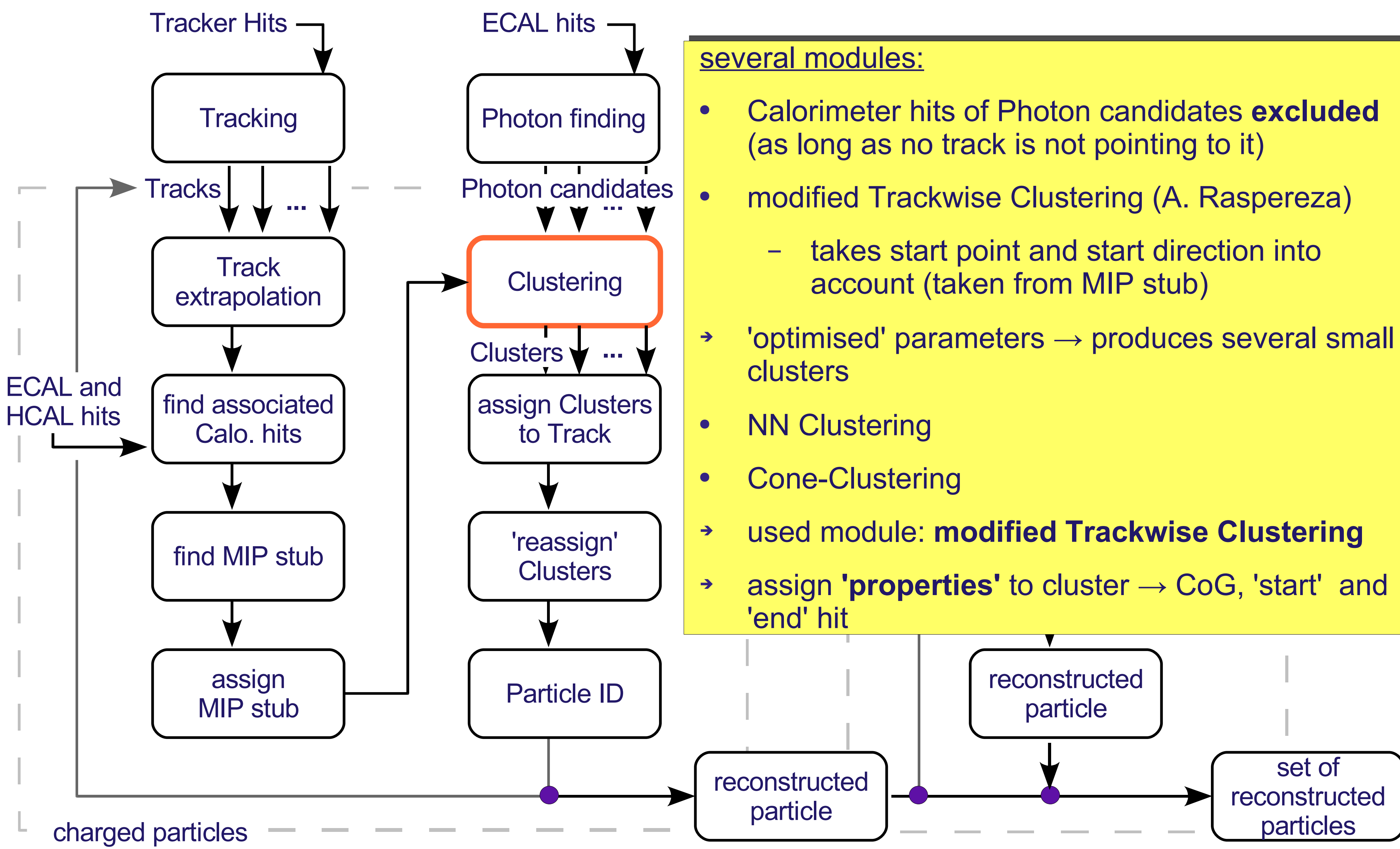


Clustering



Tracker Hits
ECAL hits
Tracking
Photon finding
Tracks
...
Photon candidates
...
neutral particles
ECAL and
HCAL hits
Track
extrapolation
Clustering
remove
'charged' hits
Clusters
...
ECAL and
HCAL hits
find associated
Calo. hits
assign Clusters
to Track
global
Clustering
Clusters
...
find MIP stub
'reassign'
Clusters
Particle ID
assign
MIP stub
Particle ID
reconstructed
particle
reconstructed
particle
set of
reconstructed
particles
charged particles

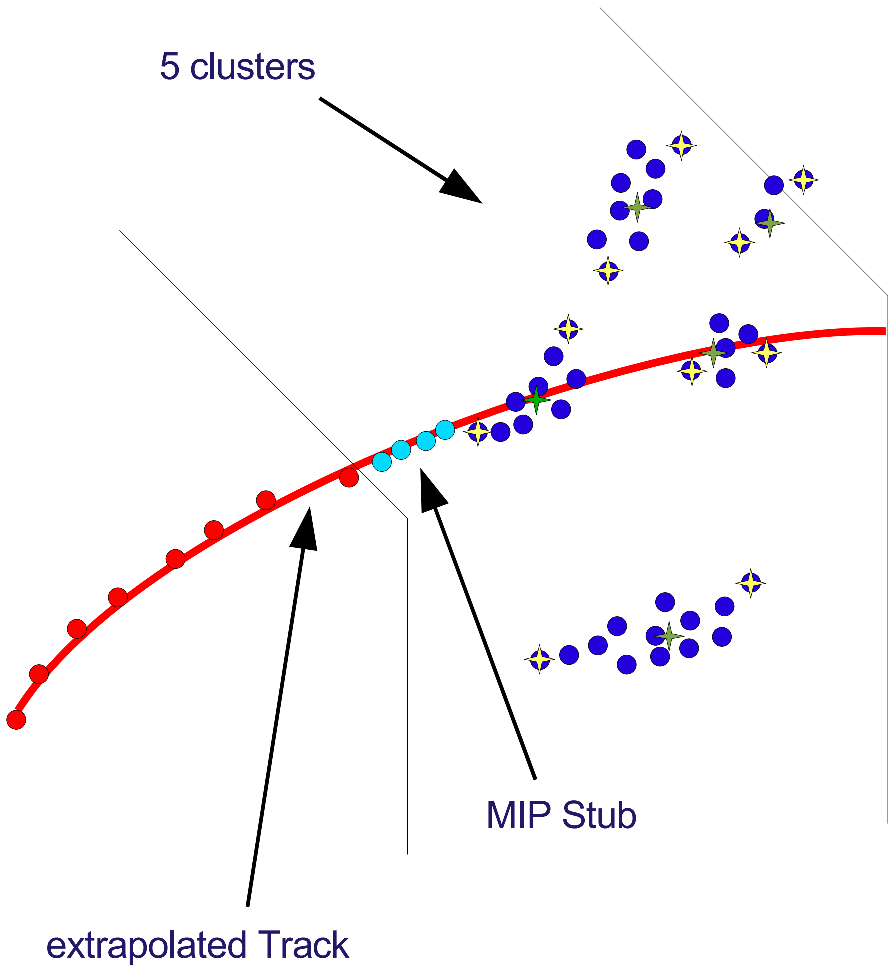
Clustering



several modules:

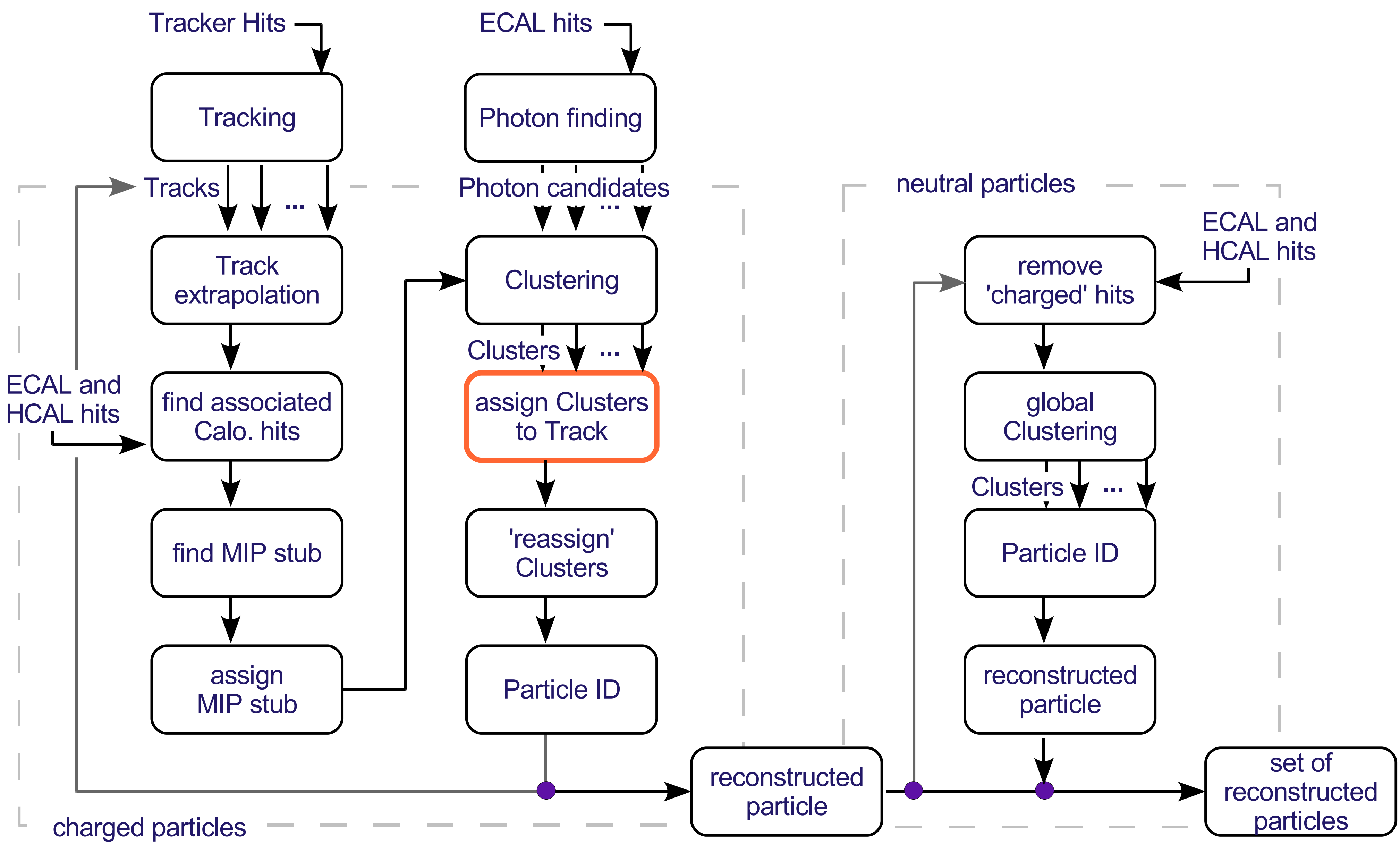
- Calorimeter hits of Photon candidates **excluded** (as long as no track is not pointing to it)
- modified Trackwise Clustering (A. Raspereza)
  - takes start point and start direction into account (taken from MIP stub)
- ➔ 'optimised' parameters → produces several small clusters
- NN Clustering
- Cone-Clustering
- ➔ used module: **modified Trackwise Clustering**
- ➔ assign **'properties'** to cluster → CoG, 'start' and 'end' hit

Properties of Clusters

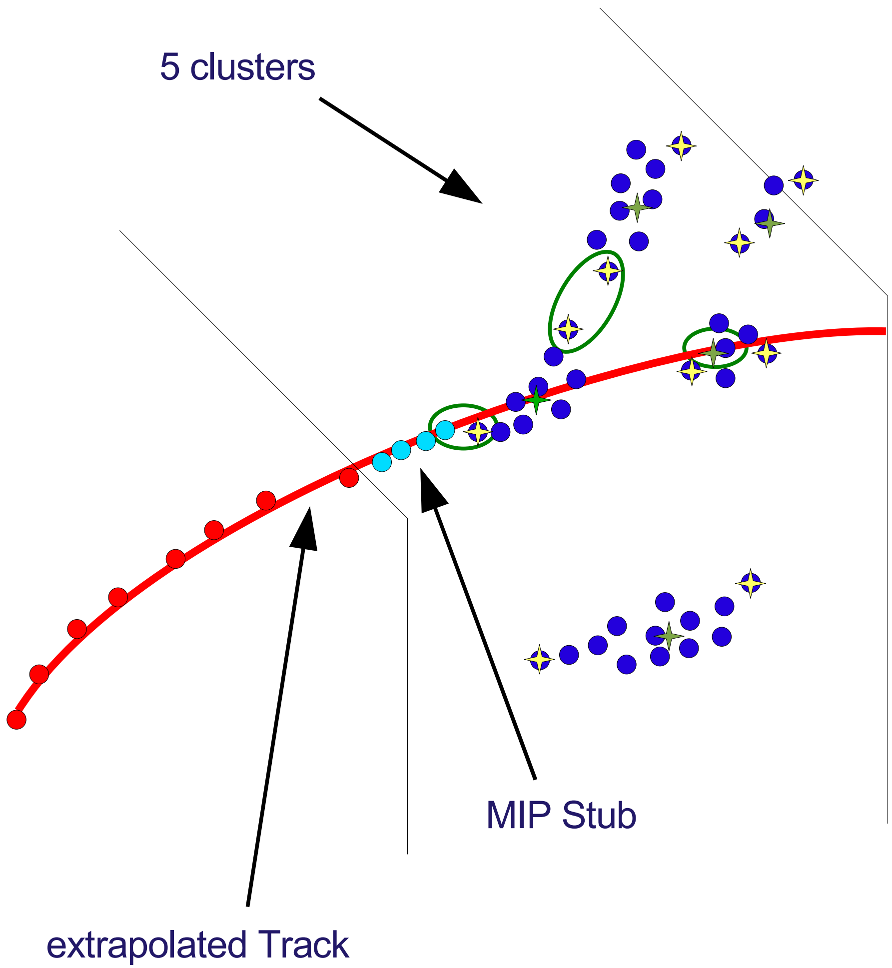


- calculate Center of Gravity (CoG) for each cluster (✦)
- calculate start and end hit for each cluster (✦)
  - simply smallest and largest path length of Calorimeter Hit on trajectory/helix
  - these properties are assigned to each cluster

Assign Clusters to Track



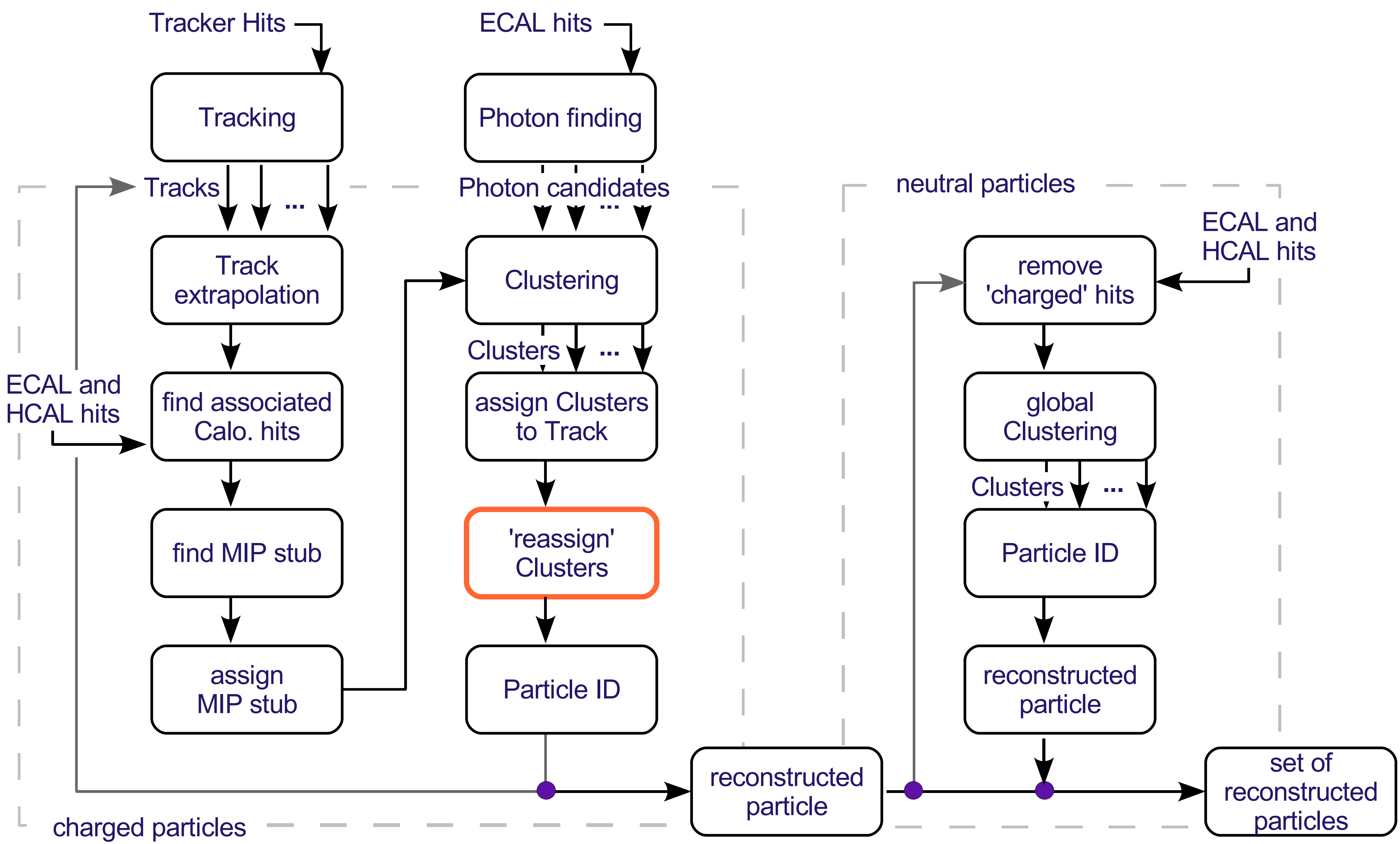
Assign Clusters to Track



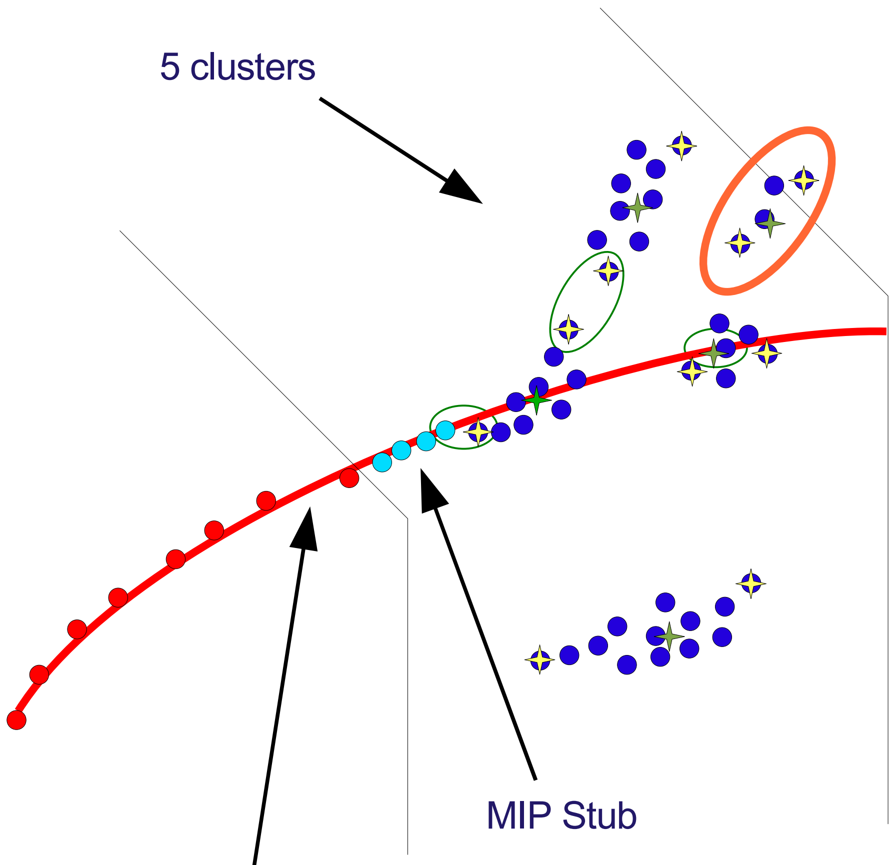
assign Cluster to track if

- distance between end point of cluster i to start point of cluster j is smaller than a given limit
- ➔ limit depends on sampling fraction
- distance of CoG to extrapolated track is smaller than a given limit
- some more geometrical conditions

'Reassign' Clusters



'Reassign' Clusters

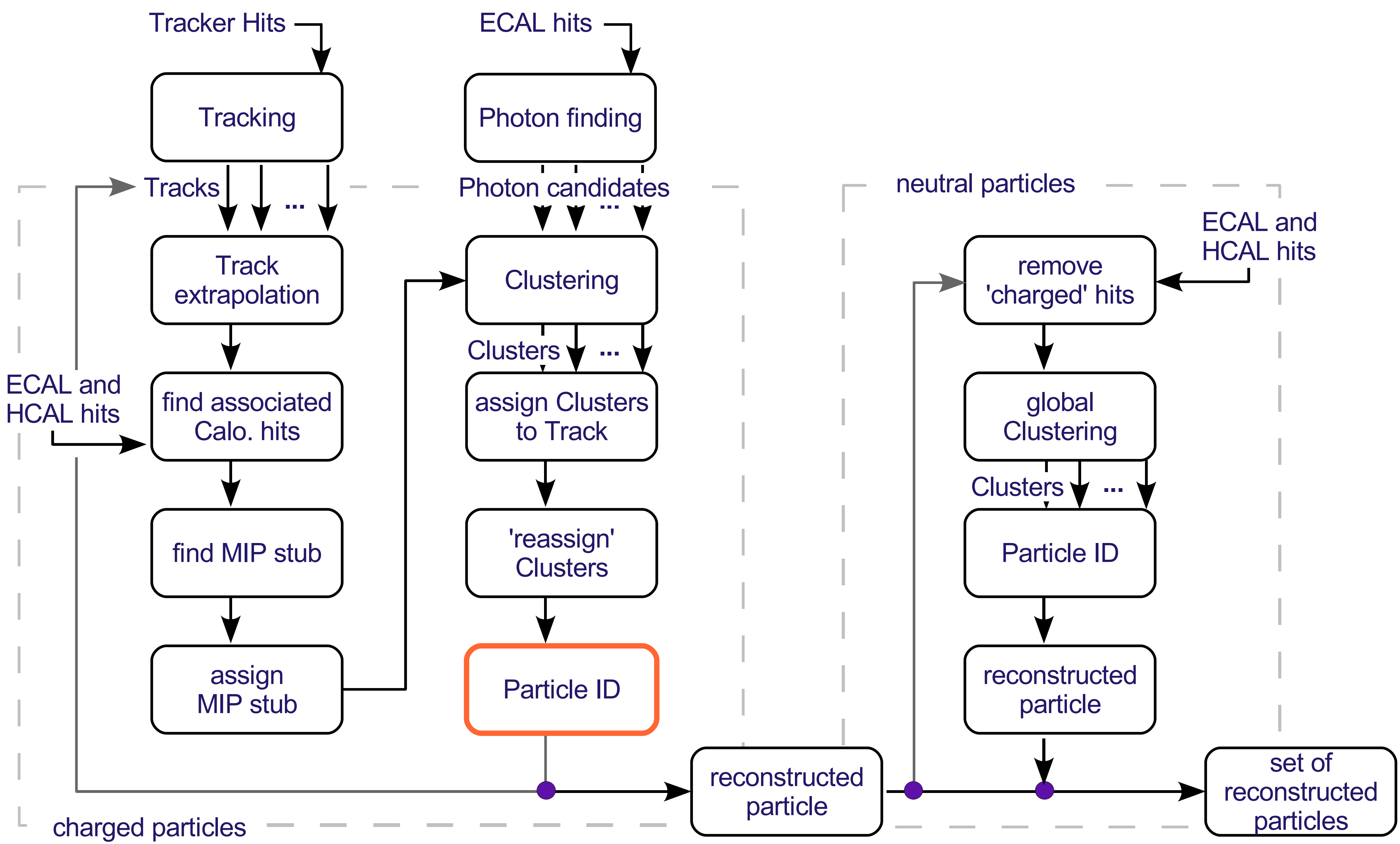


assign additional Clusters if

- distance of cluster is smaller than a certain limit
- $|E_1 + E_2 - E_{Track}| < 3\ \sigma_E$

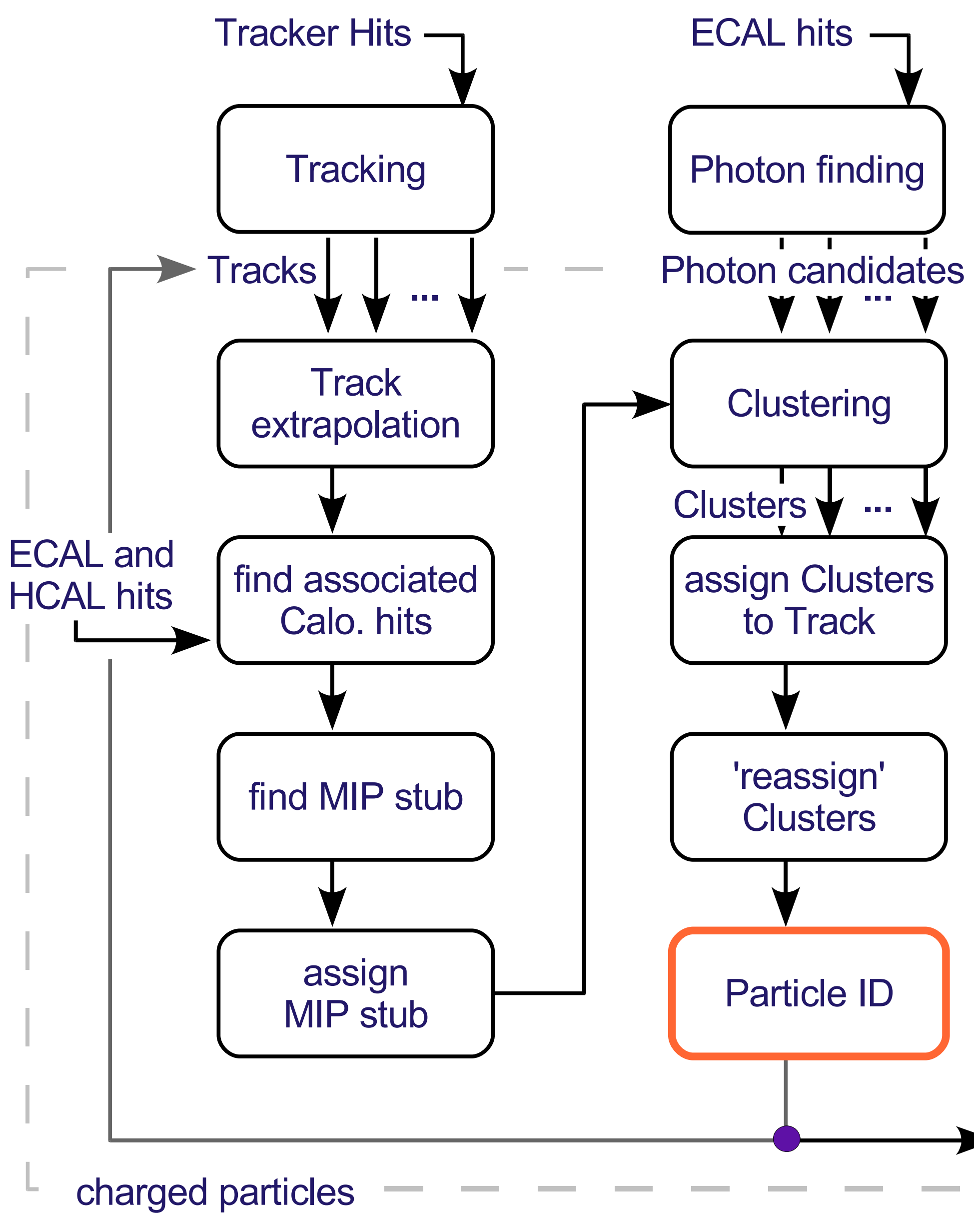
→ very simple at the moment

Particle ID



Tracker Hits
ECAL hits
Tracking
Photon finding
Tracks
...
Photon candidates
...
neutral particles
ECAL and HCAL hits
Track extrapolation
Clustering
remove 'charged' hits
Clusters
...
ECAL and HCAL hits
find associated Calo. hits
assign Clusters to Track
global Clustering
Clusters
...
find MIP stub
'reassign' Clusters
Particle ID
assign MIP stub
Particle ID
reconstructed particle
reconstructed particle
set of reconstructed particles
charged particles

Particle ID



very simple procedure at the moment (adapted from Wolf):

- fraction of energy ECAL/HCAL, given by a steering parameter
  - if fraction > 95 % → particle is an electron
  - if fraction < 95 % → particle is a pion

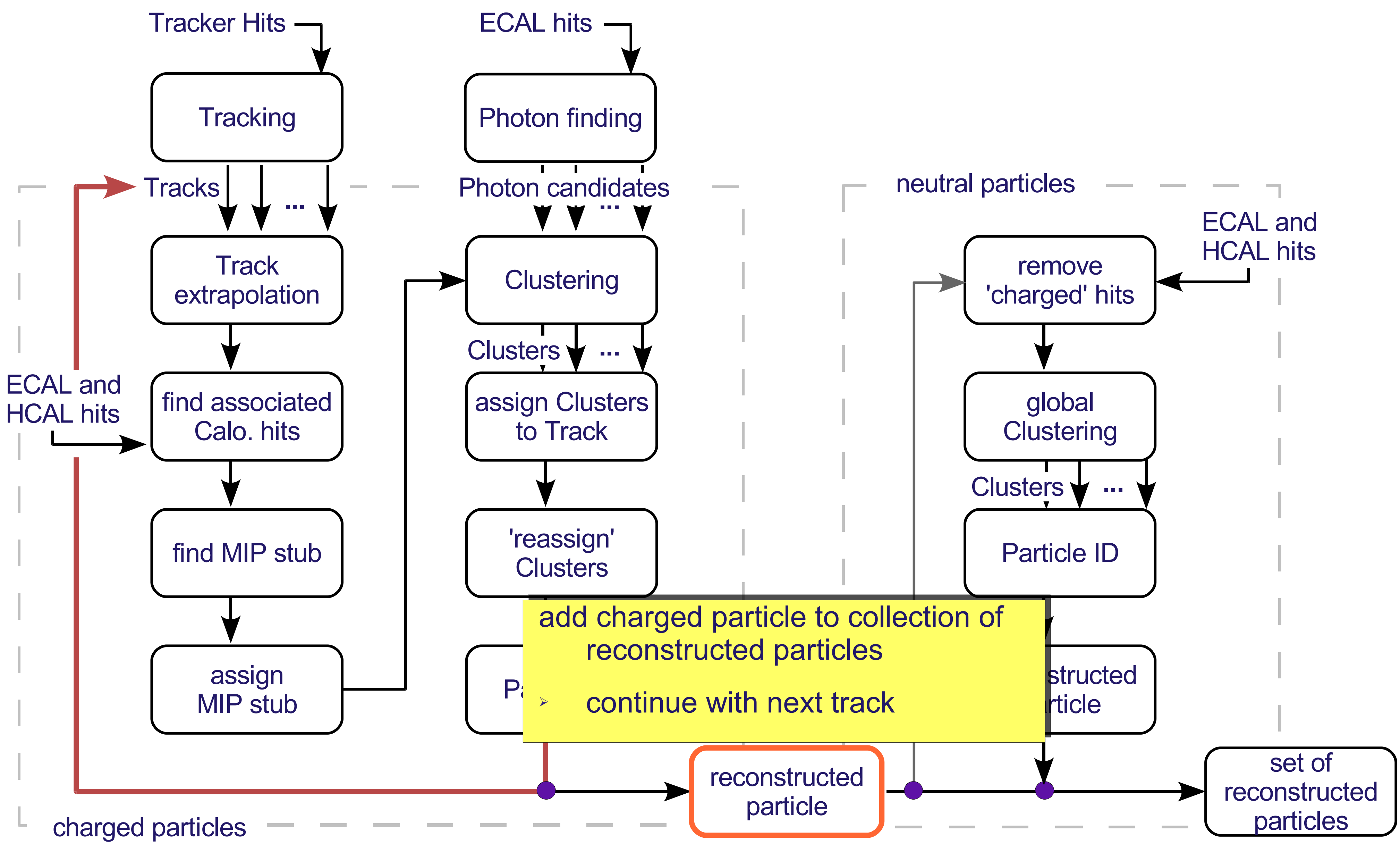
additionally:

- if there is only the MIP stub assigned to the track → particle is a muon

a lot of room for improvements:

- fit of the e.m. Shower Shape
- do 'real' electron (find 'photon' with corresponding track) and muon finding

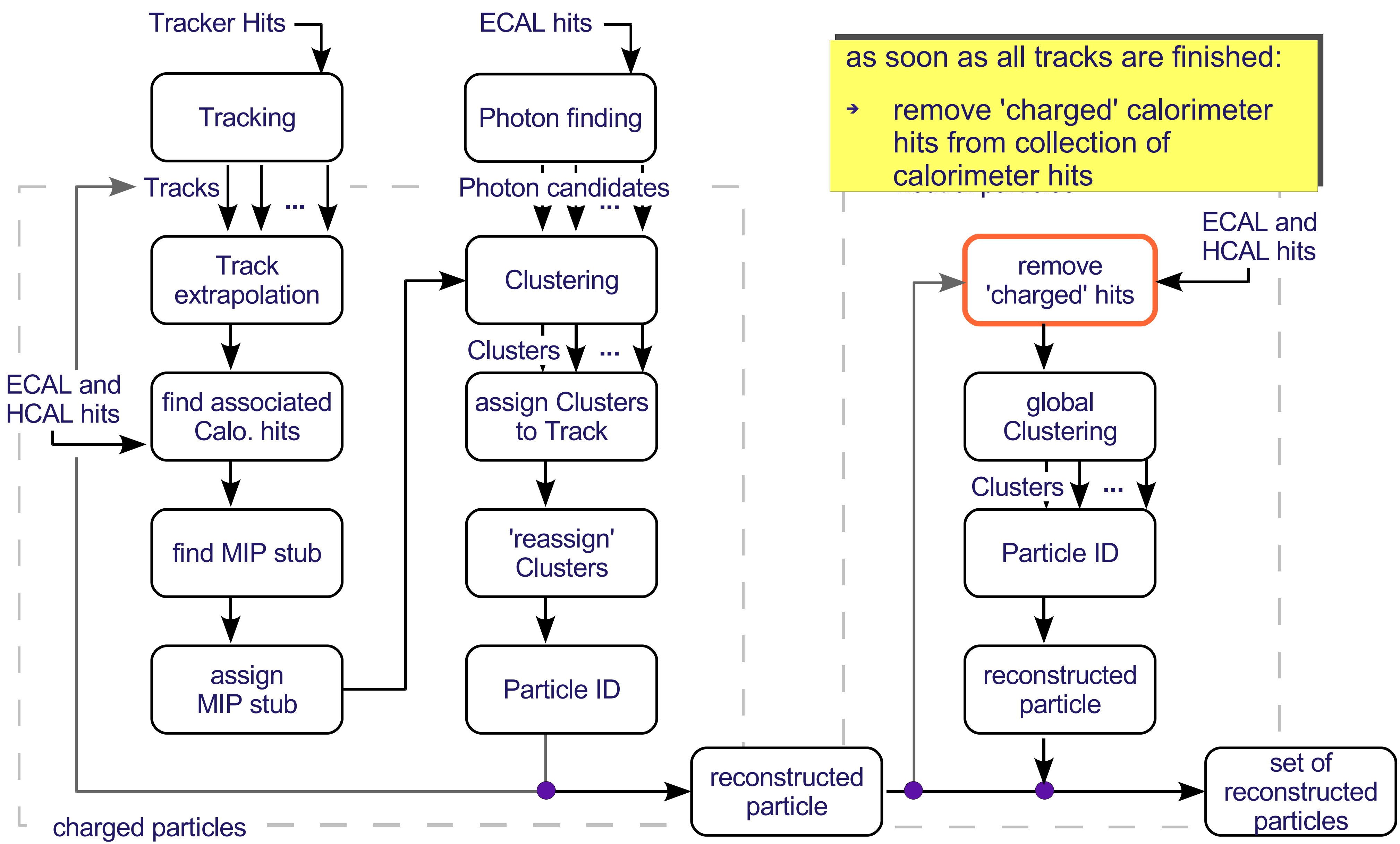
Charged Particle reconstructed



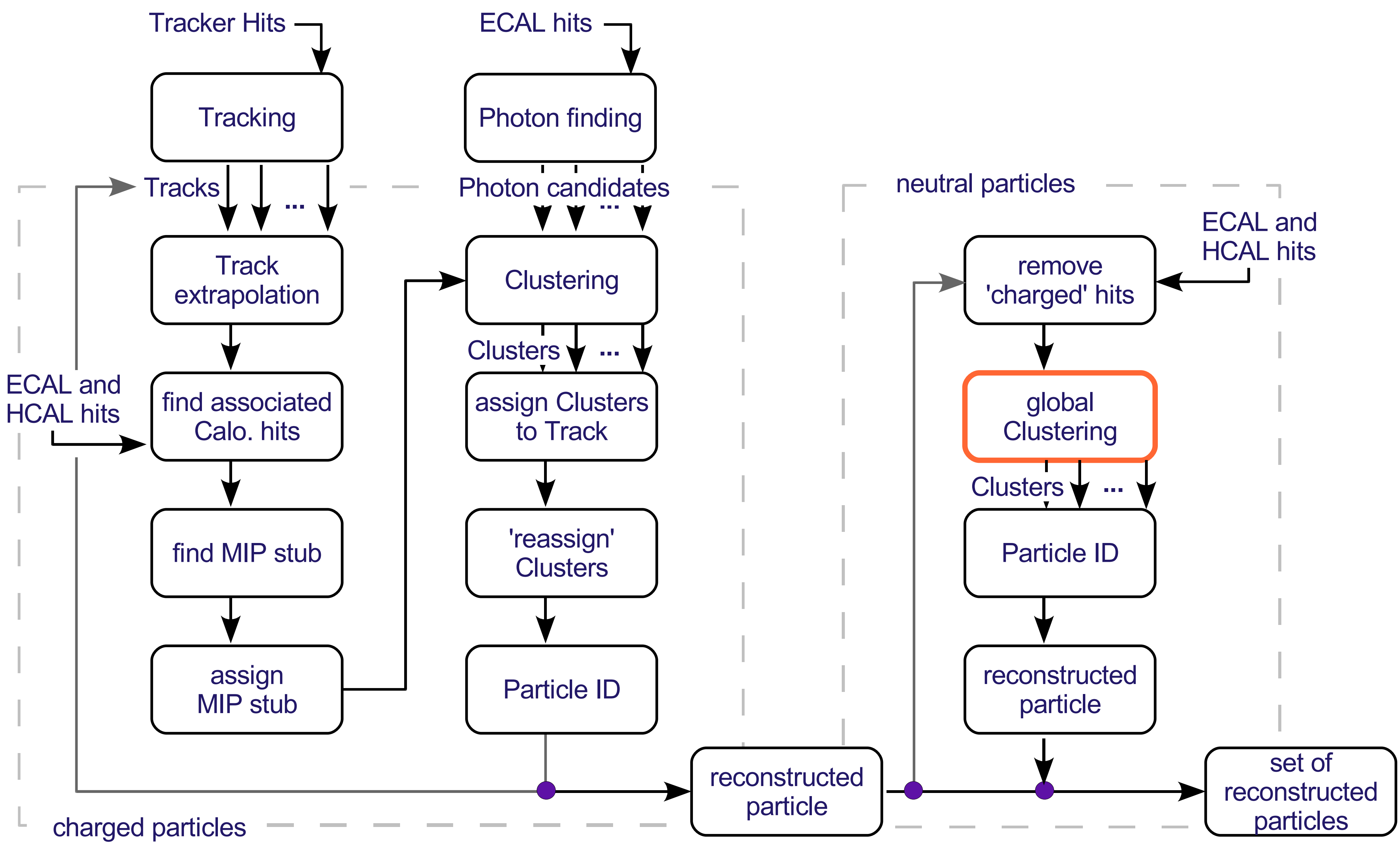
Remove 'Charged' Calorimeter Hits

as soon as all tracks are finished:

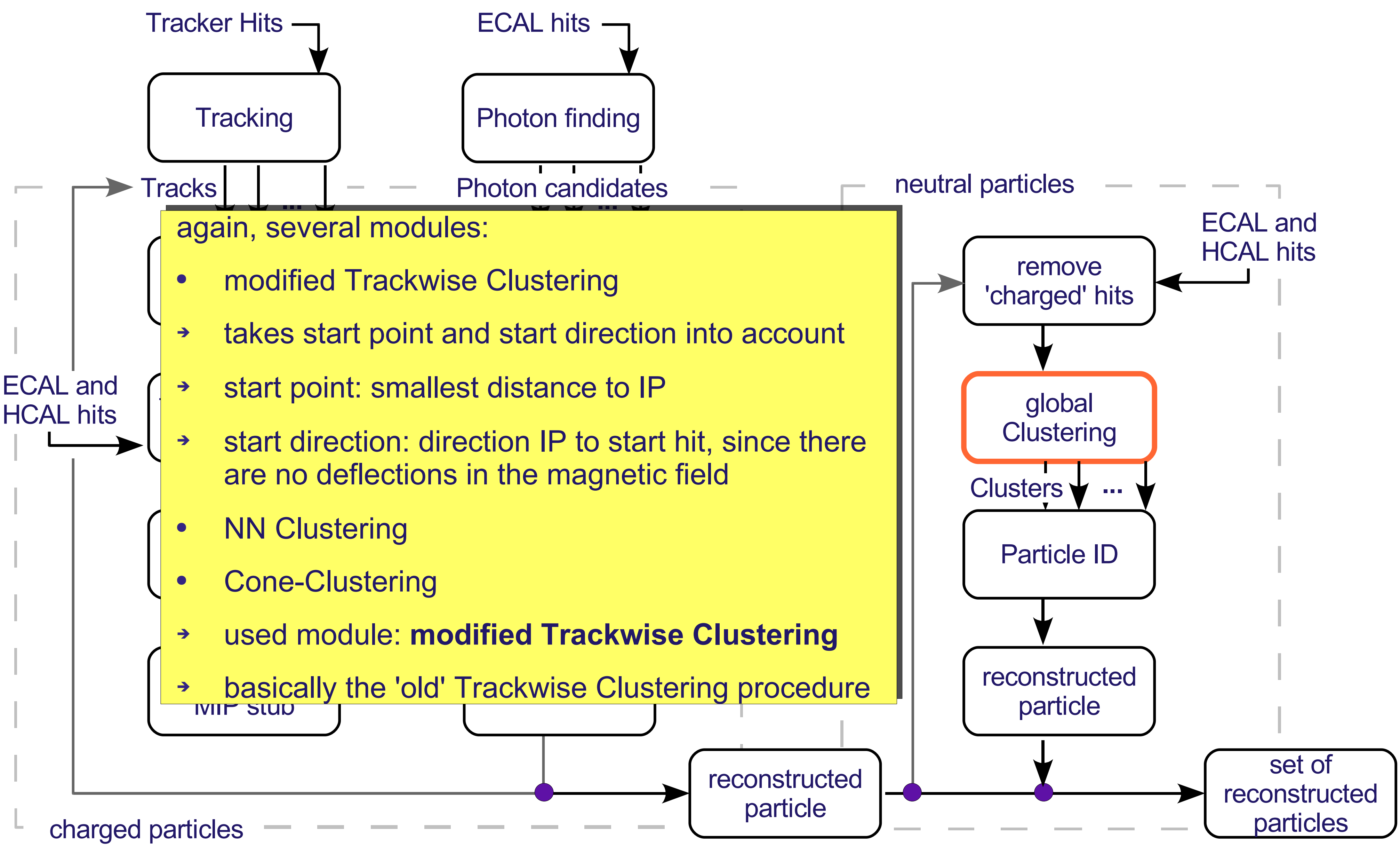
- remove 'charged' calorimeter hits from collection of calorimeter hits



Clustering on Neutral Hits



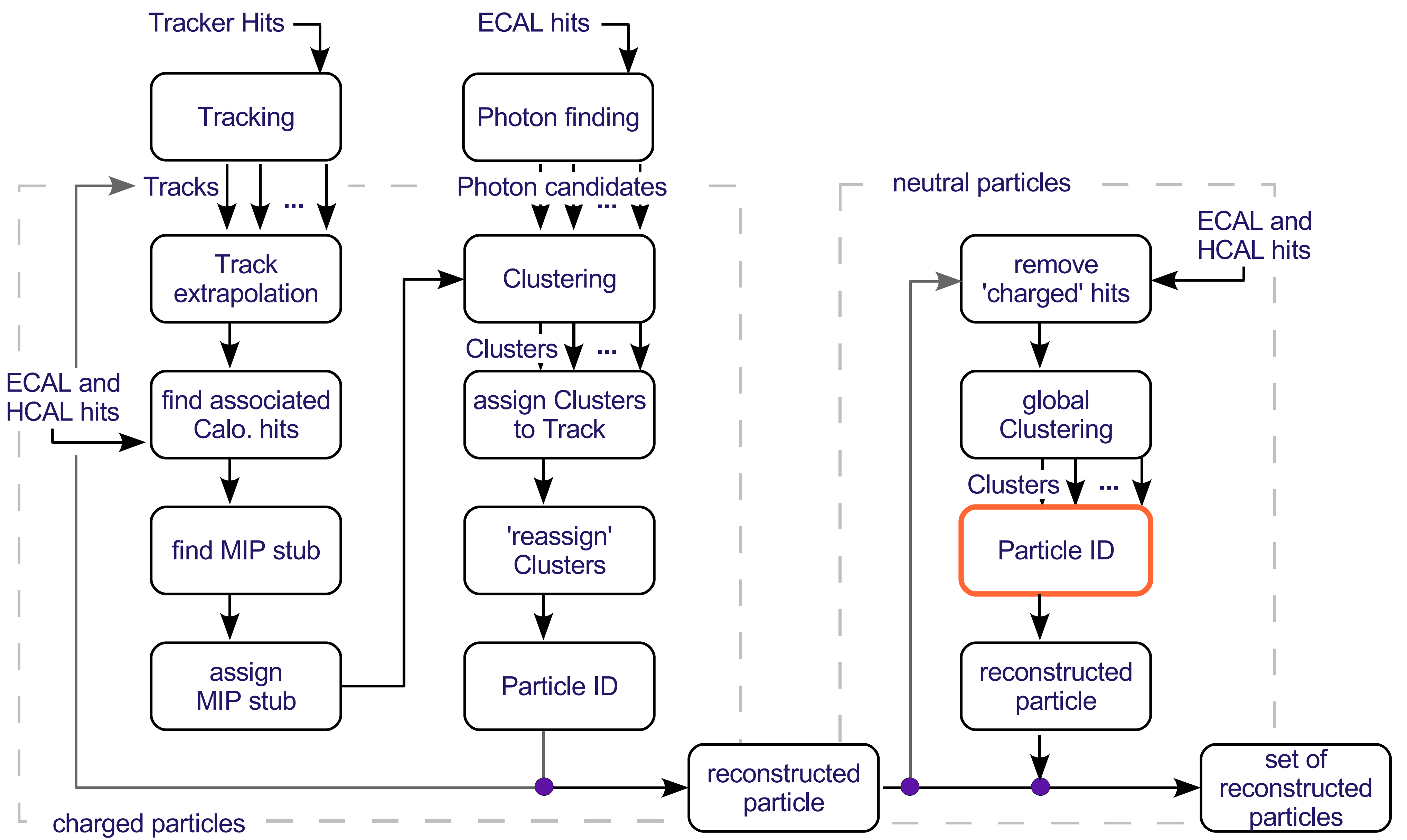
Clustering on Neutral Hits



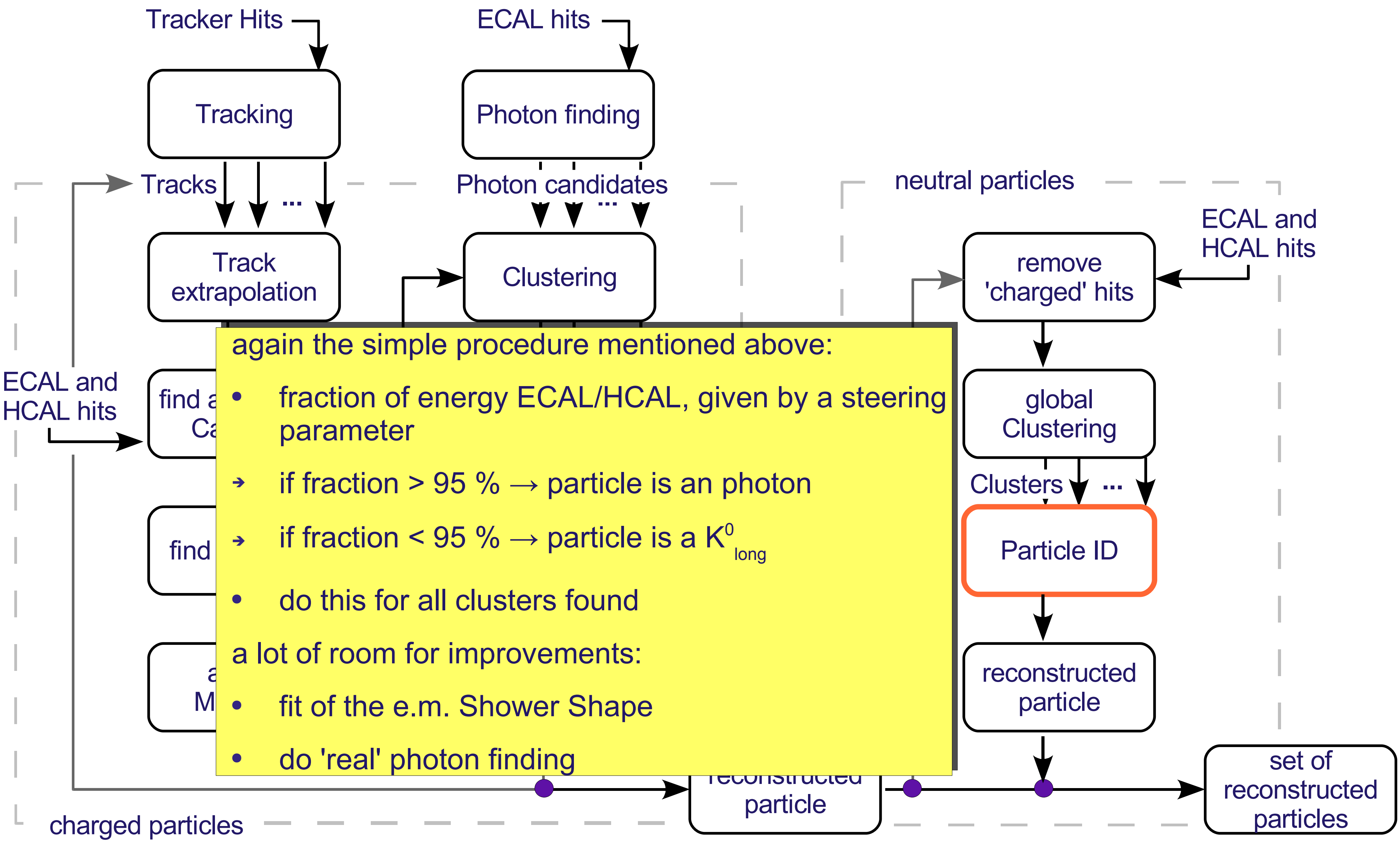
again, several modules:

- modified Trackwise Clustering
  - takes start point and start direction into account
  - start point: smallest distance to IP
  - start direction: direction IP to start hit, since there are no deflections in the magnetic field
- NN Clustering
- Cone-Clustering
  - used module: **modified Trackwise Clustering**
  - basically the 'old' Trackwise Clustering procedure

Particle ID for Neutrals



Particle ID for Neutrals



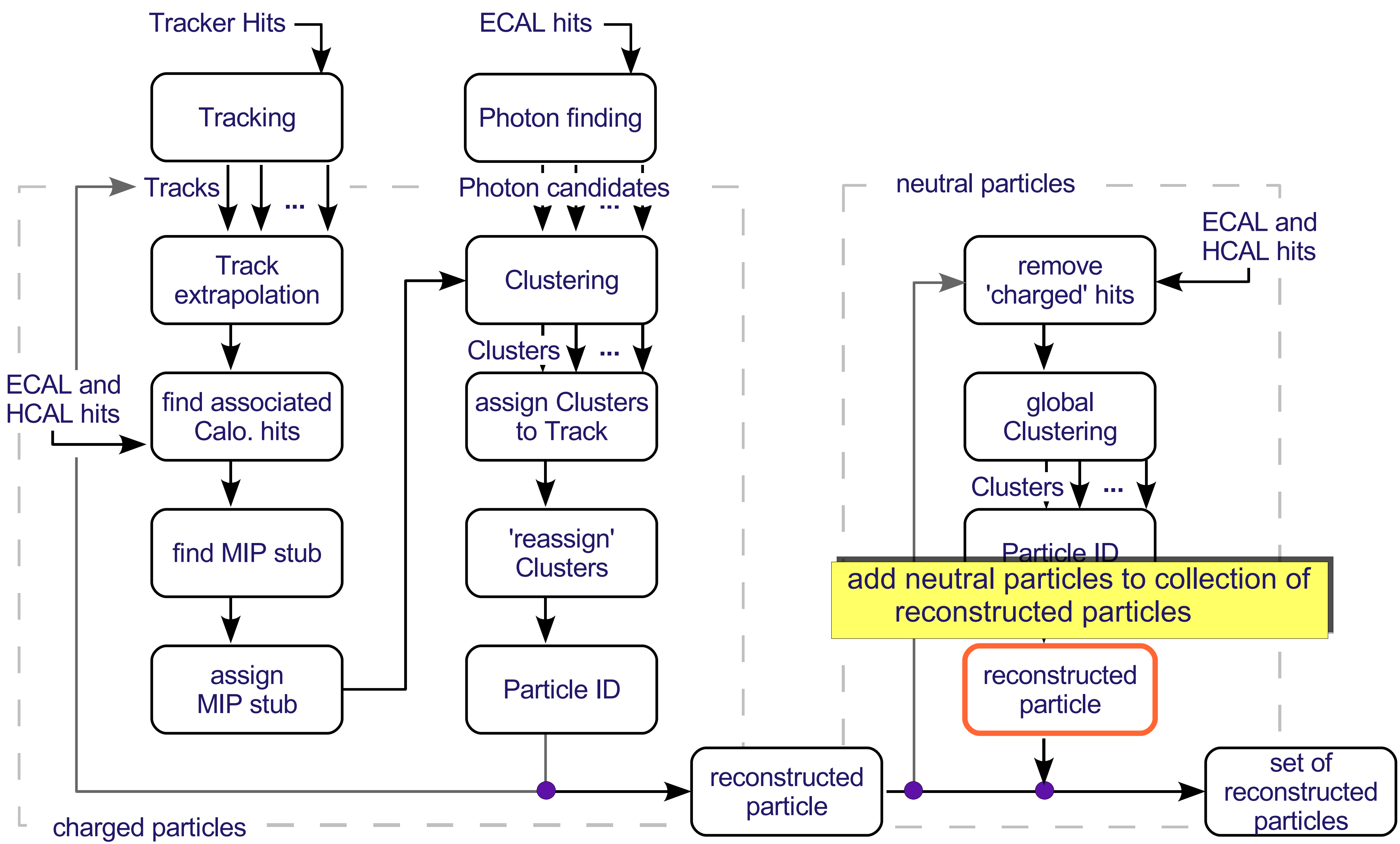
again the simple procedure mentioned above:

- fraction of energy ECAL/HCAL, given by a steering parameter
  - if fraction > 95 % → particle is an photon
  - if fraction < 95 % → particle is a K^0_{long}
- do this for all clusters found

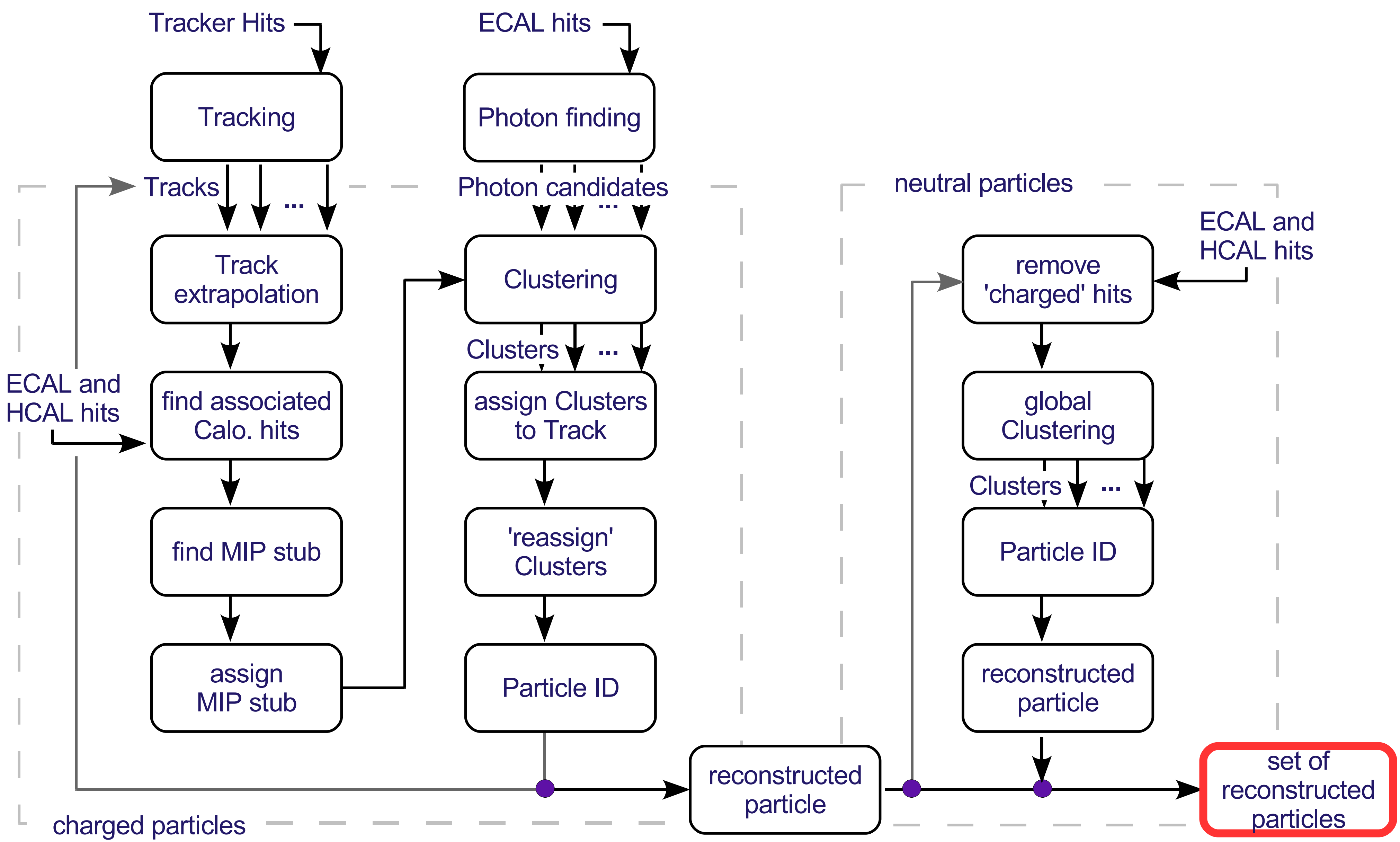
a lot of room for improvements:

- fit of the e.m. Shower Shape
- do 'real' photon finding

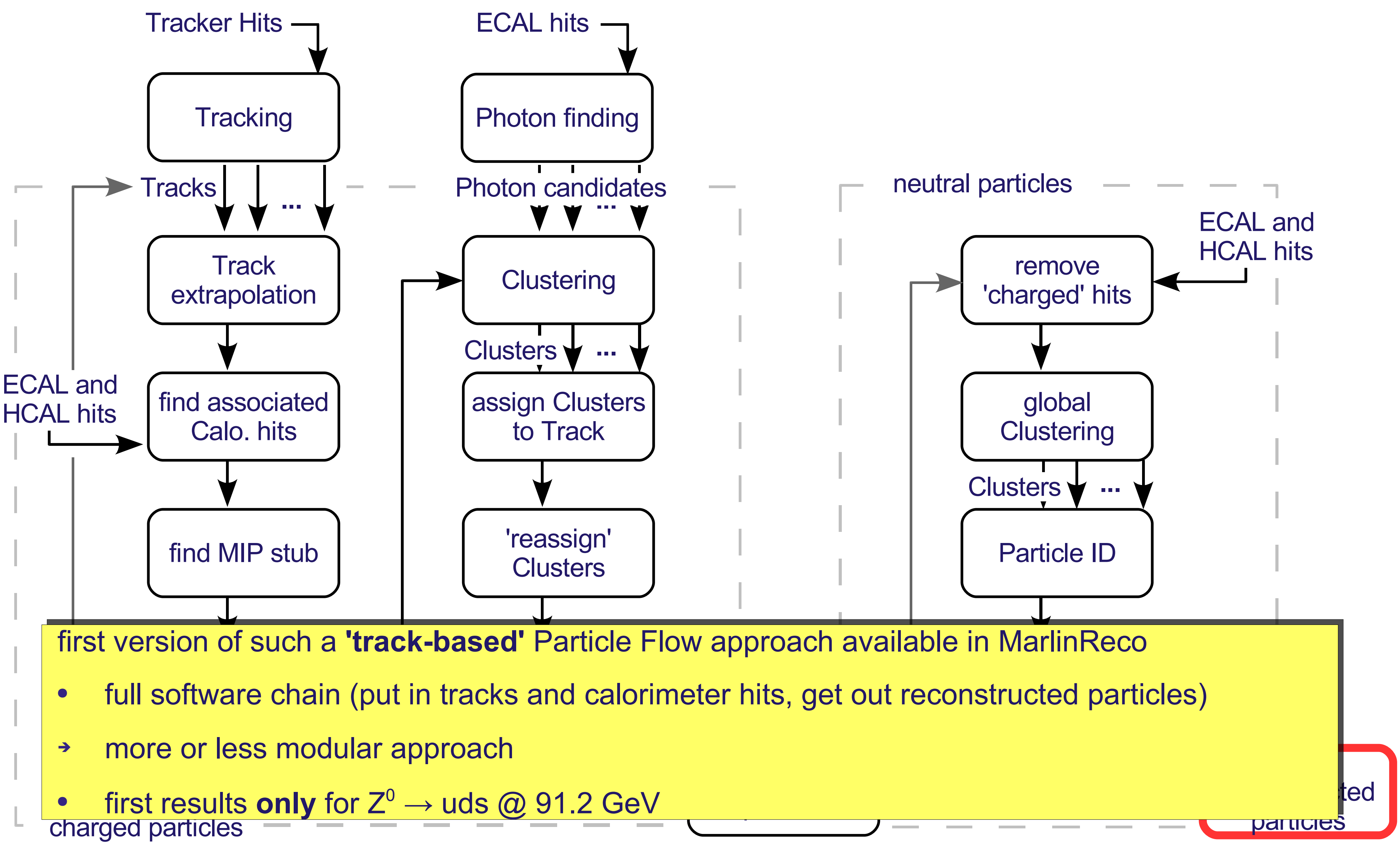
Neutral Particle reconstructed



End of Track-Based Particle Flow Algorithm



Track-Based Particle Flow Algorithm



first version of such a **'track-based'** Particle Flow approach available in MarlinReco

- full software chain (put in tracks and calorimeter hits, get out reconstructed particles)
- → more or less modular approach
- first results **only** for $Z^0 \rightarrow$ uds @ 91.2 GeV

Performance of Track-Based PFlow

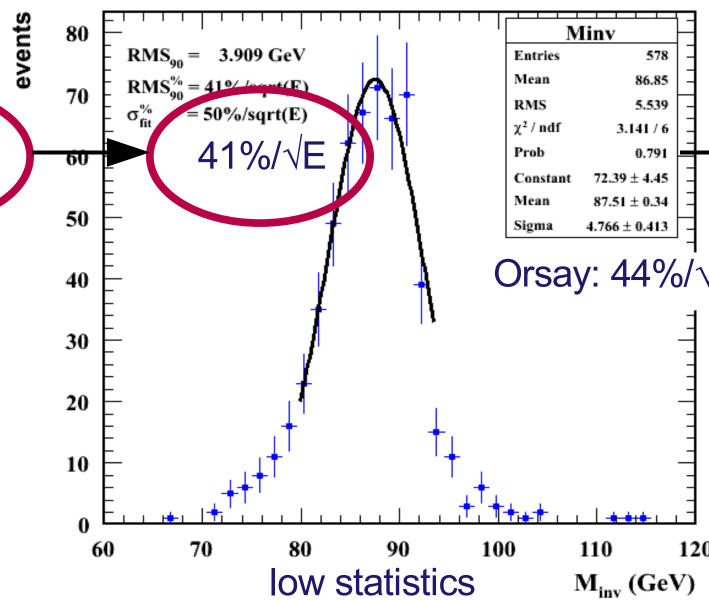
<u>some first results</u> for Z → uds @ 91.2 GeV, cos(θ) < 0.8, LDC00Sc, R(1690mm), L(2730mm):

Wolf

52%/√E

realistic **Track-Based PFlow**

41%/√E



low statistics

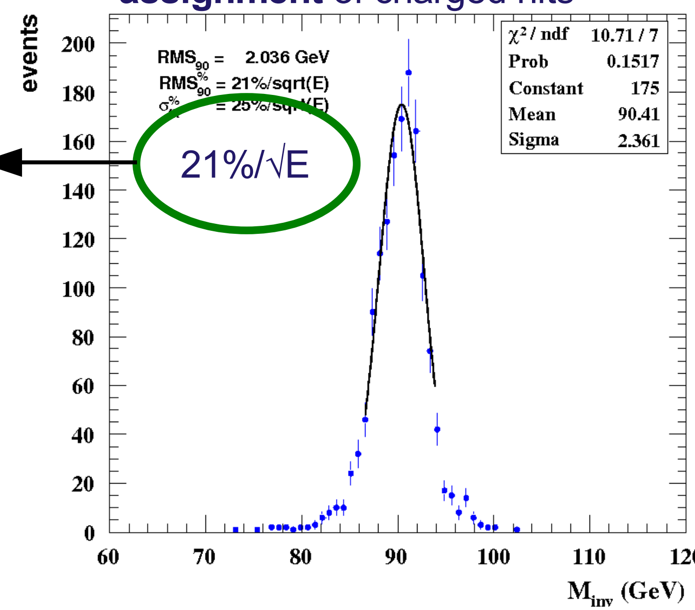
Orsay: 44%/√E

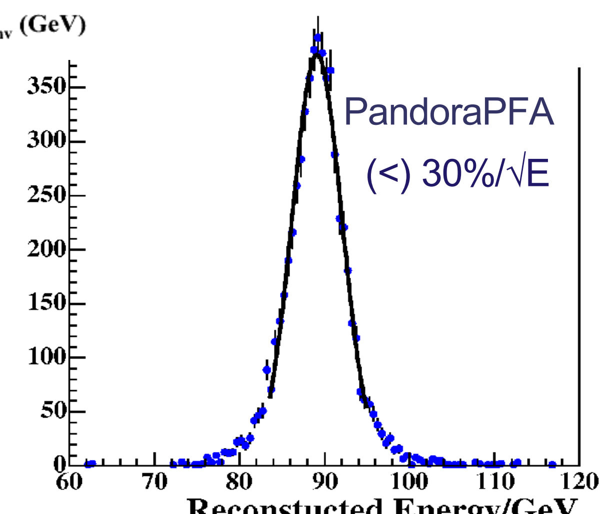
GOAL

30%/√E

Track-Based PFlow with **perfect assignment** of charged hits

21%/√E





PandoraPFA

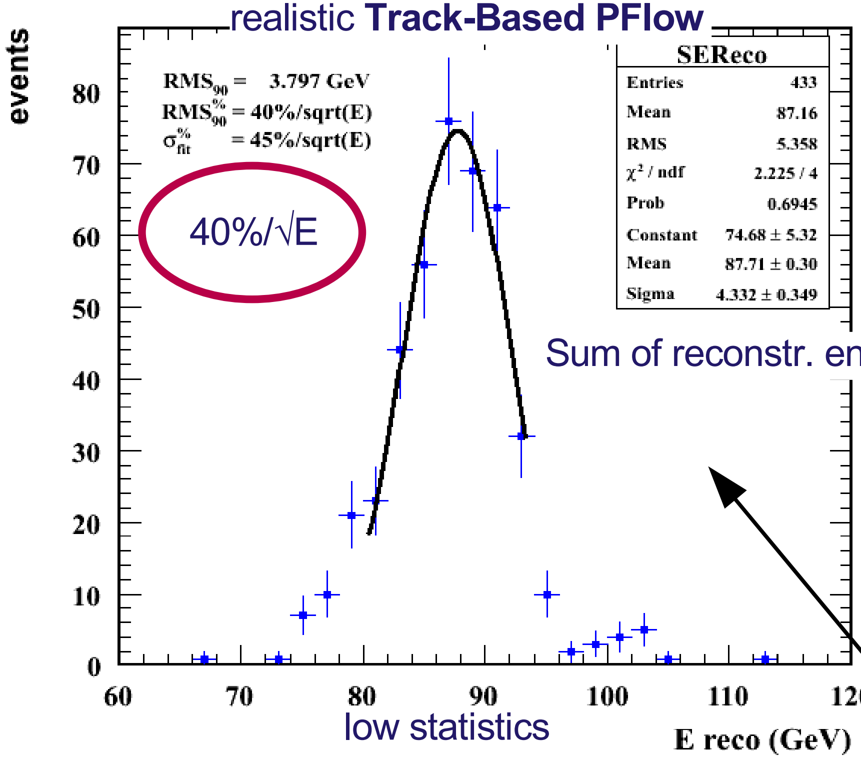
(<) 30%/√E

- **exceeds performance** of Wolf but still (significantly) worse than PandoraPFA
- perfect assignment comes close to 'theoretical' limit
- → no major bugs in the code

Performance of Track-Based PFlow

some first results for Z → uds @ 91.2 GeV, cos(θ) < 0.8, LDC00Sc, R(1690mm), L(2730mm):

realistic **Track-Based PFlow**



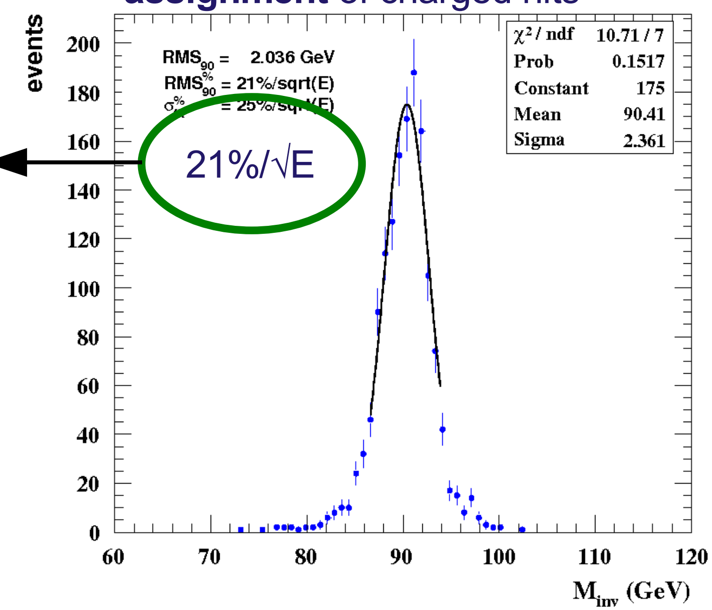
Sum of reconstr. energy

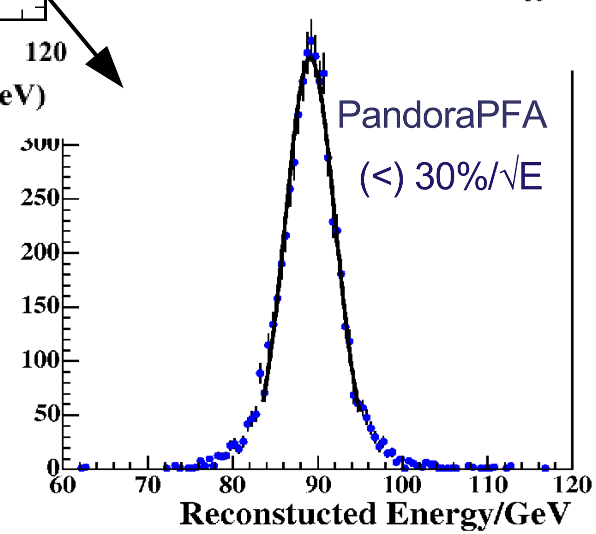
low statistics

GOAL

30%/√E

Track-Based PFlow with **perfect assignment** of charged hits





PandoraPFA

(<) 30%/√E

- **exceeds performance** of Wolf but still (significantly) worse than PandoraPFA
- perfect assignment comes close to 'theoretical' limit

➔ no major bugs in the code

Conclusions and Outlook

Conclusions:

- Track-Based Particle Flow is evolving, **improvements** made since Valencia/Orsay
- performance **exceeds** performance of Wolf but still (significantly) worse than PandoraPFA
  - → algorithm is far perfect and not optimised in any direction (processing time)
  - → relatively 'young' approach, work in ongoing
- new modules (Photon finding and cluster 'reassignment') and bug-fixes since Orsay
- **initial** version of a Track-Based Particle Flow **available** in Marlin (in MarlinReco cvs)

→ first attempt to use it for higher energies (M. Faucci Giannelli and K. Wichmann)

- gain understanding of 'intrinsic' problems / properties of Particle Flow algorithms in general

➢ need **multiple/different** Particle Flow approaches to **disentangle** detector and algorithm effects on the reconstruction/physics performance

Outlook:

- go for $30\%/\sqrt{E_{jet}}$ @ 91.2 GeV first, then address higher energies
- follow the path given by PandoraPFA, do detector optimisation studies and **compare results**
- study **sub-structure** of hadronic showers (e.m., hadr. part) in Calice physics prototype
  - → apply results to 'clustering' of hadronic energy in the Track-Bases Particle Flow

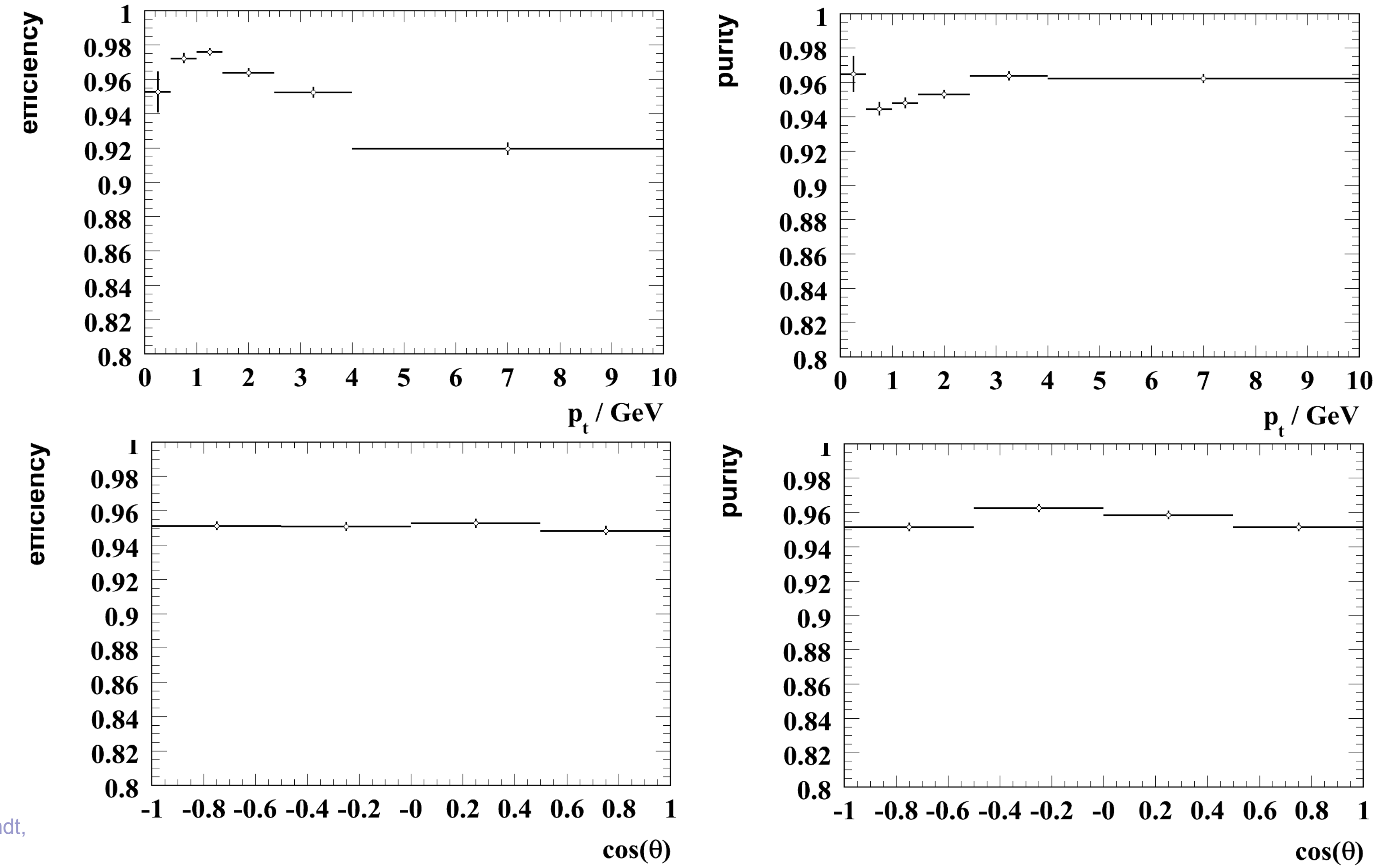
inputs / ideas / discussions are welcome

backup slides ...

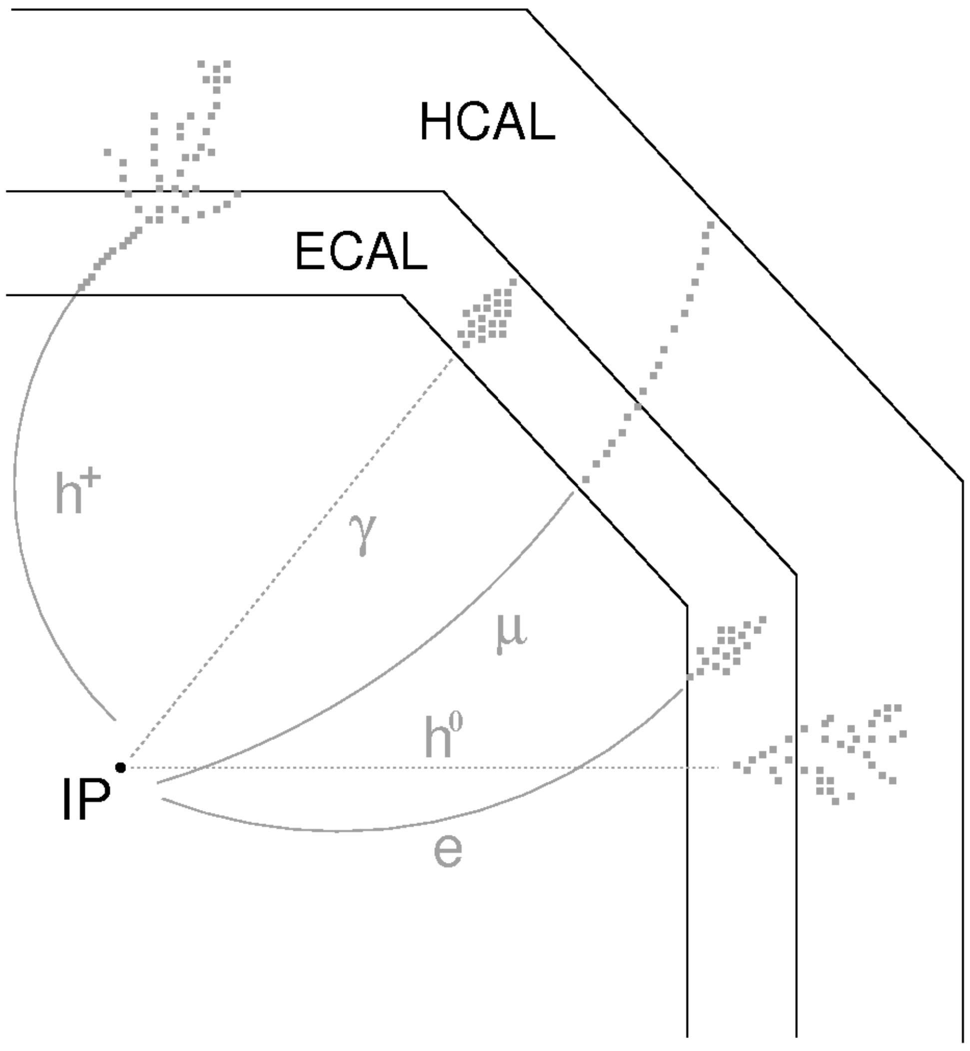
Details on MIP stub finding

Comparison with MC:

- efficiency and purity vs. p_t and cos(θ):
- overall efficiency >90%, overall purity >90%

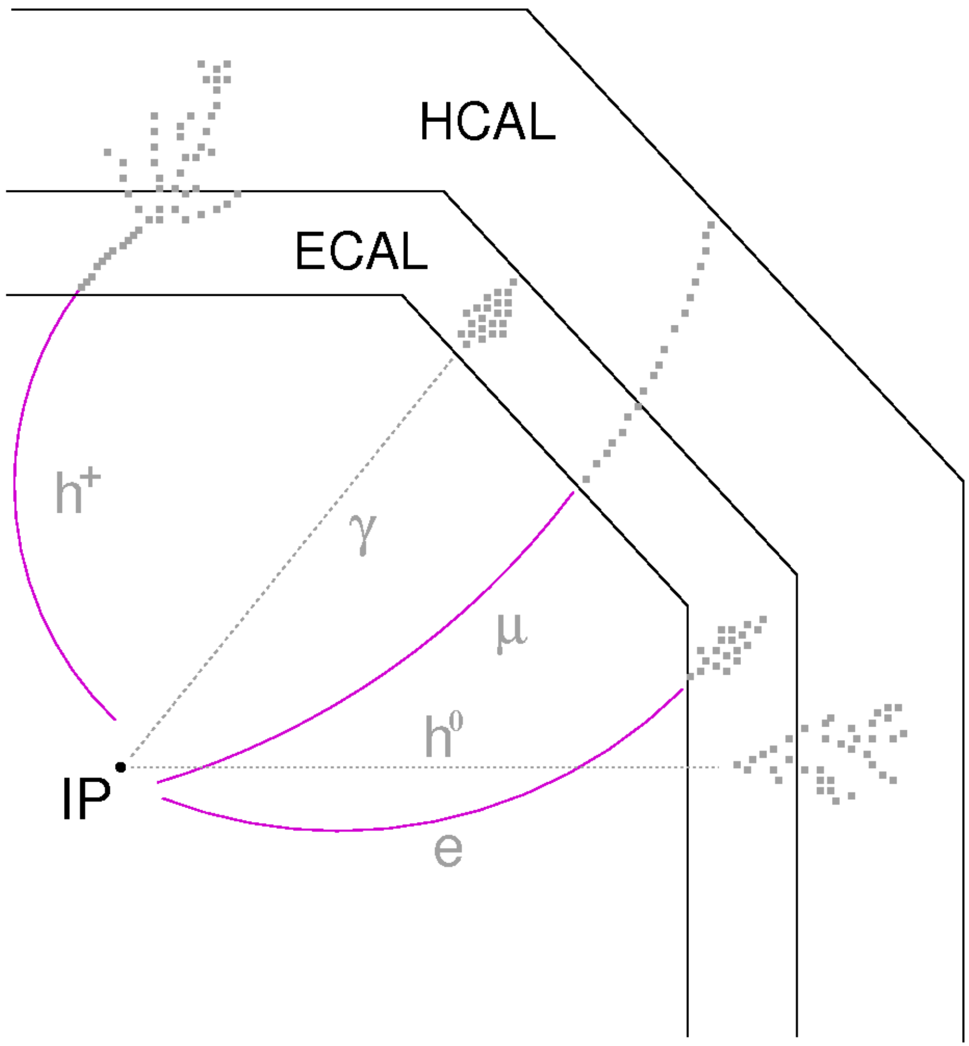


Track-Based Particle Flow Concept



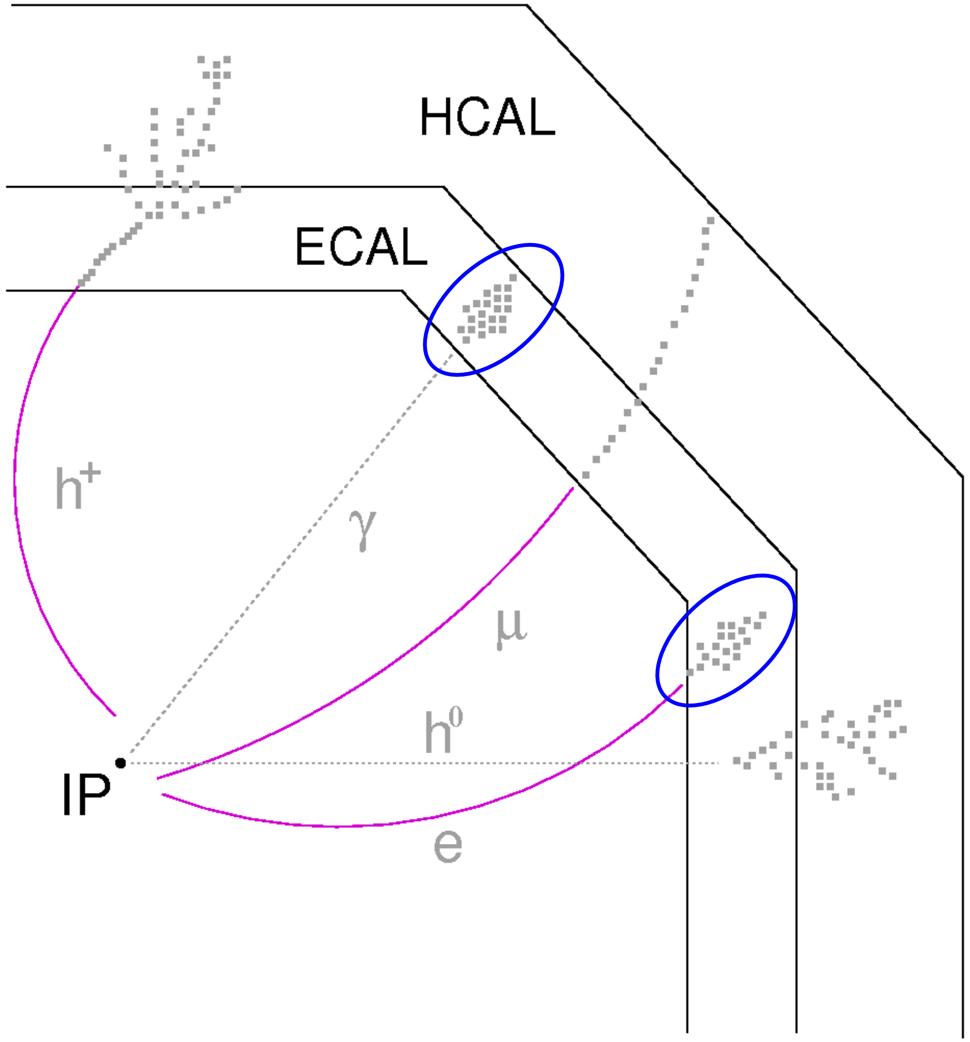
Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)



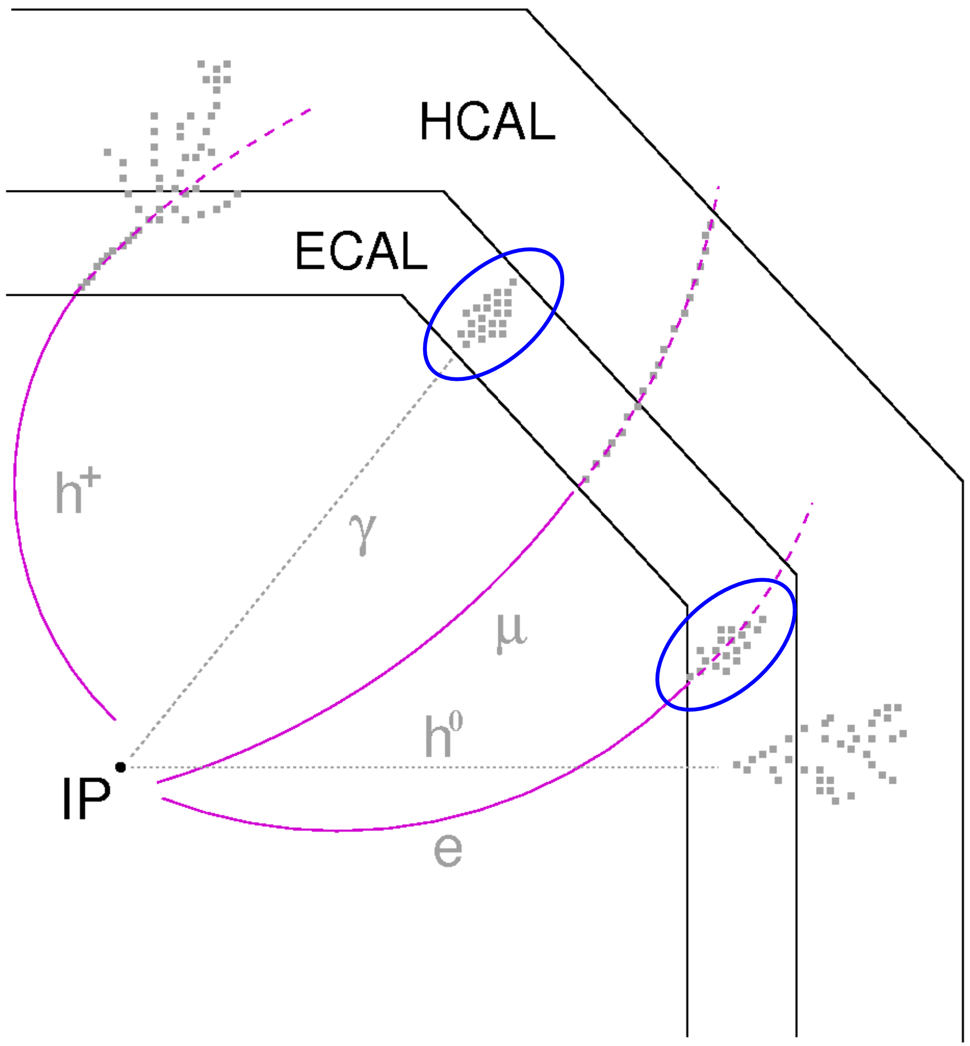
Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)
2. find photon candidates



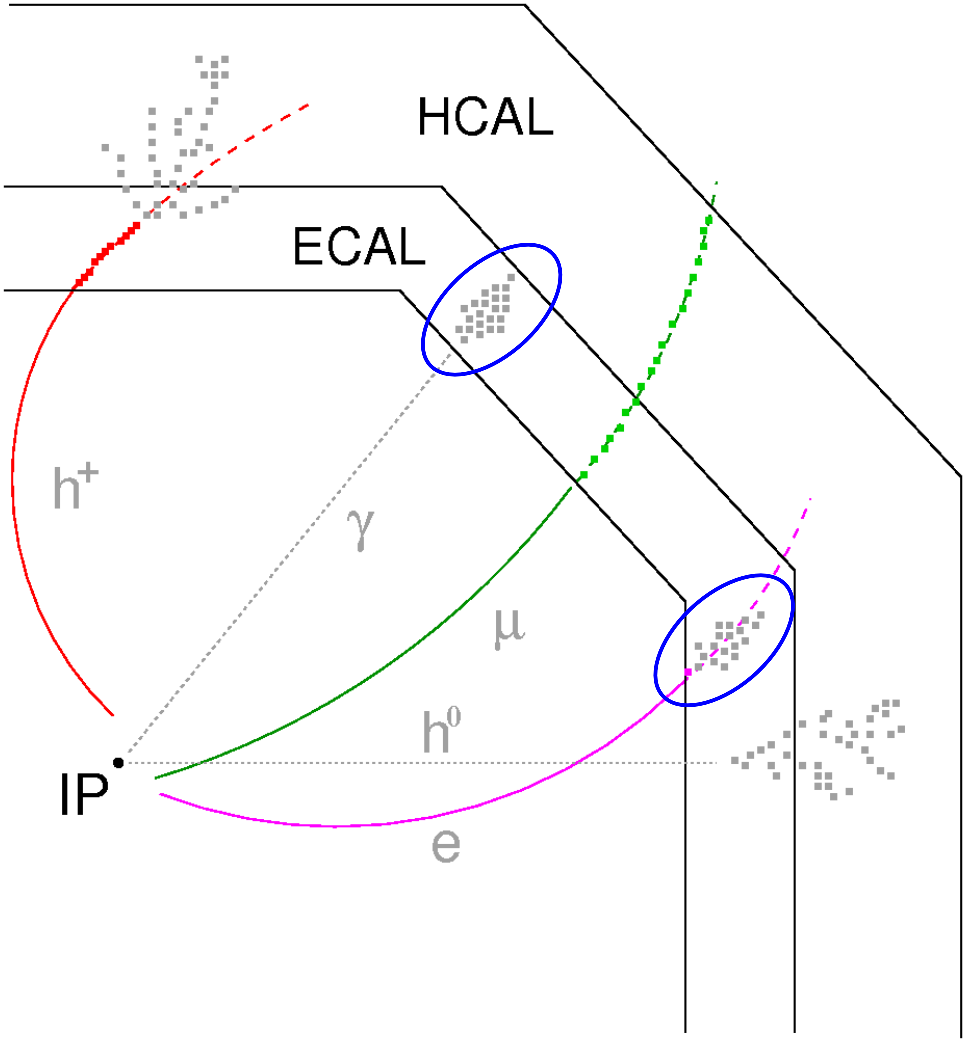
Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)
2. find photon candidates
3. extrapolate tracks into Calorimeter
   - different models, with and w/o energy loss, multiple scattering, ...
   - dedicated Geometry description needed



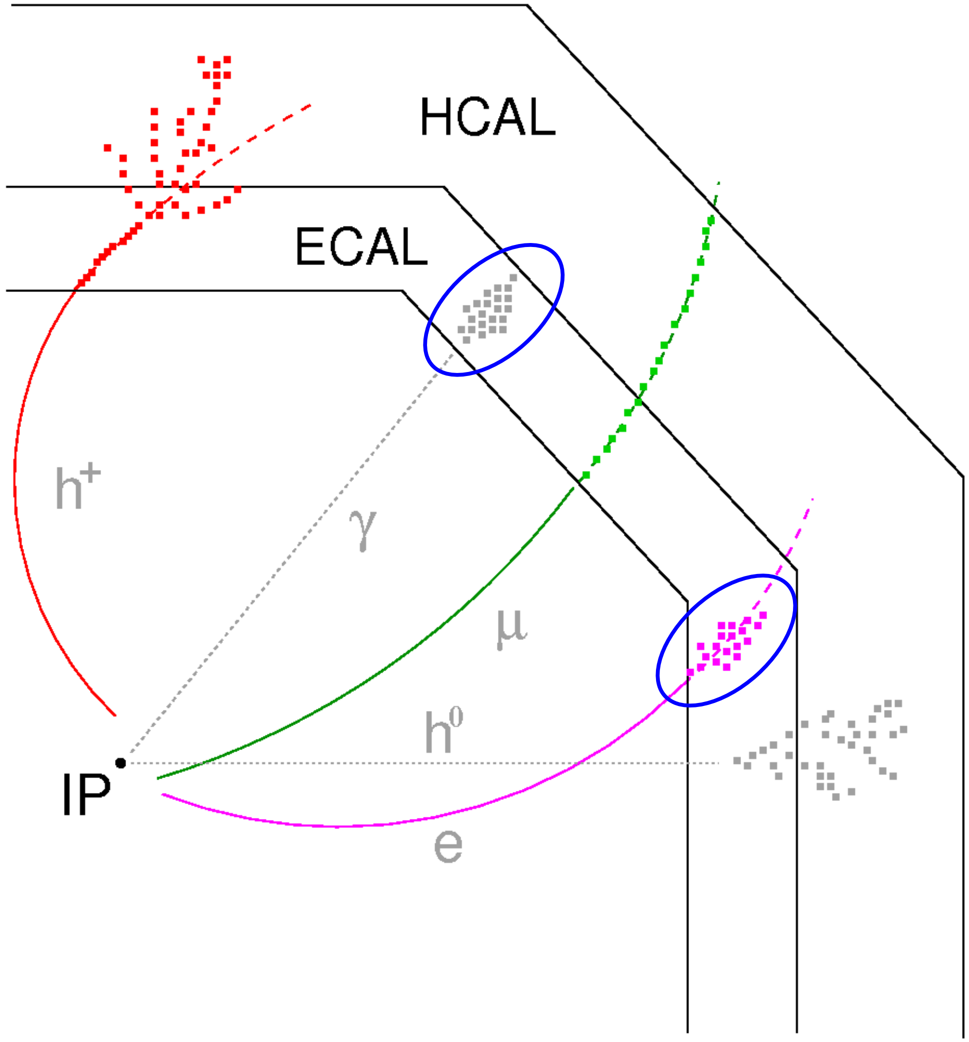
Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)
2. find photon candidates
3. extrapolate tracks into Calorimeter
   - different models, with and w/o energy loss, multiple scattering, ...
   - dedicated Geometry description needed
4. assign MIP stub to track, find muons



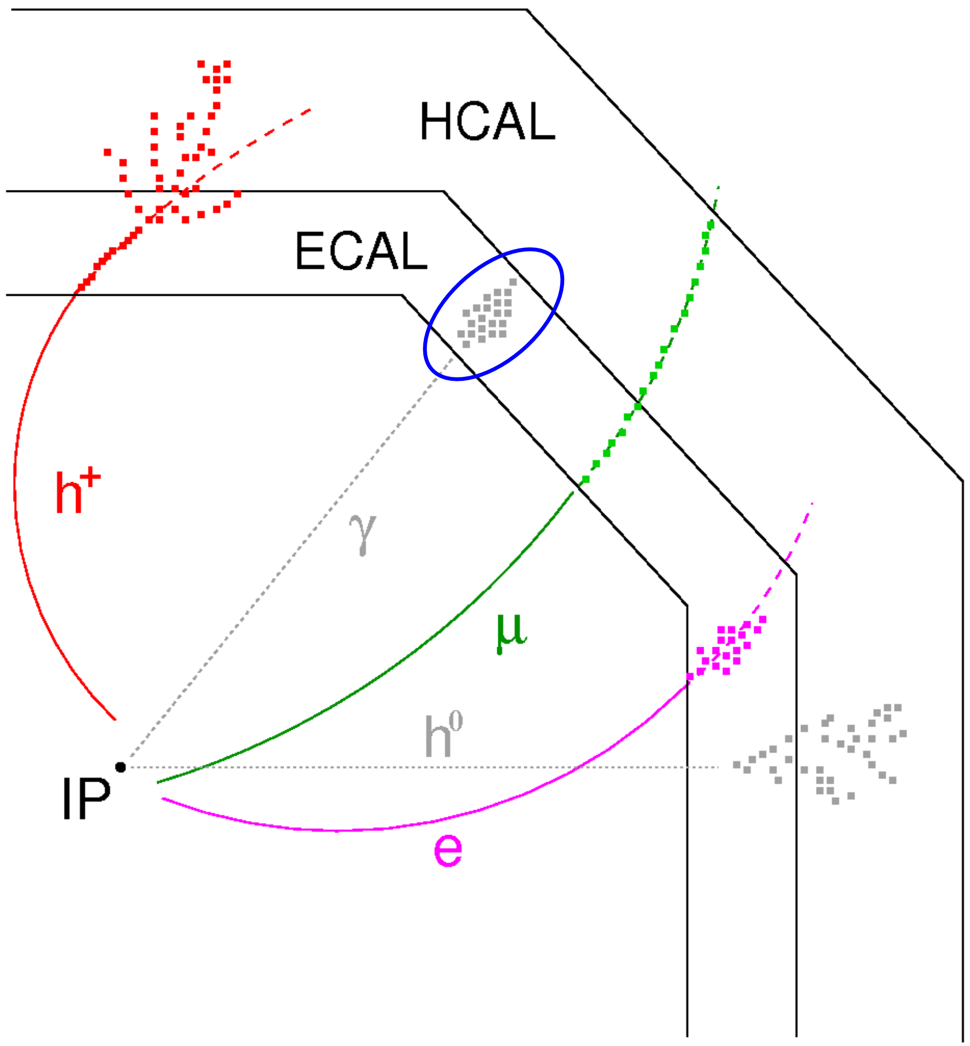
Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)
2. find photon candidates
3. extrapolate tracks into Calorimeter
   - different models, with and w/o energy loss, multiple scattering, ...
   - dedicated Geometry description needed
4. assign MIP stub to track, find muons
5. clustering (ECAL and HCAL)
   - variable, depending on track and photon candidates
   - different algorithms



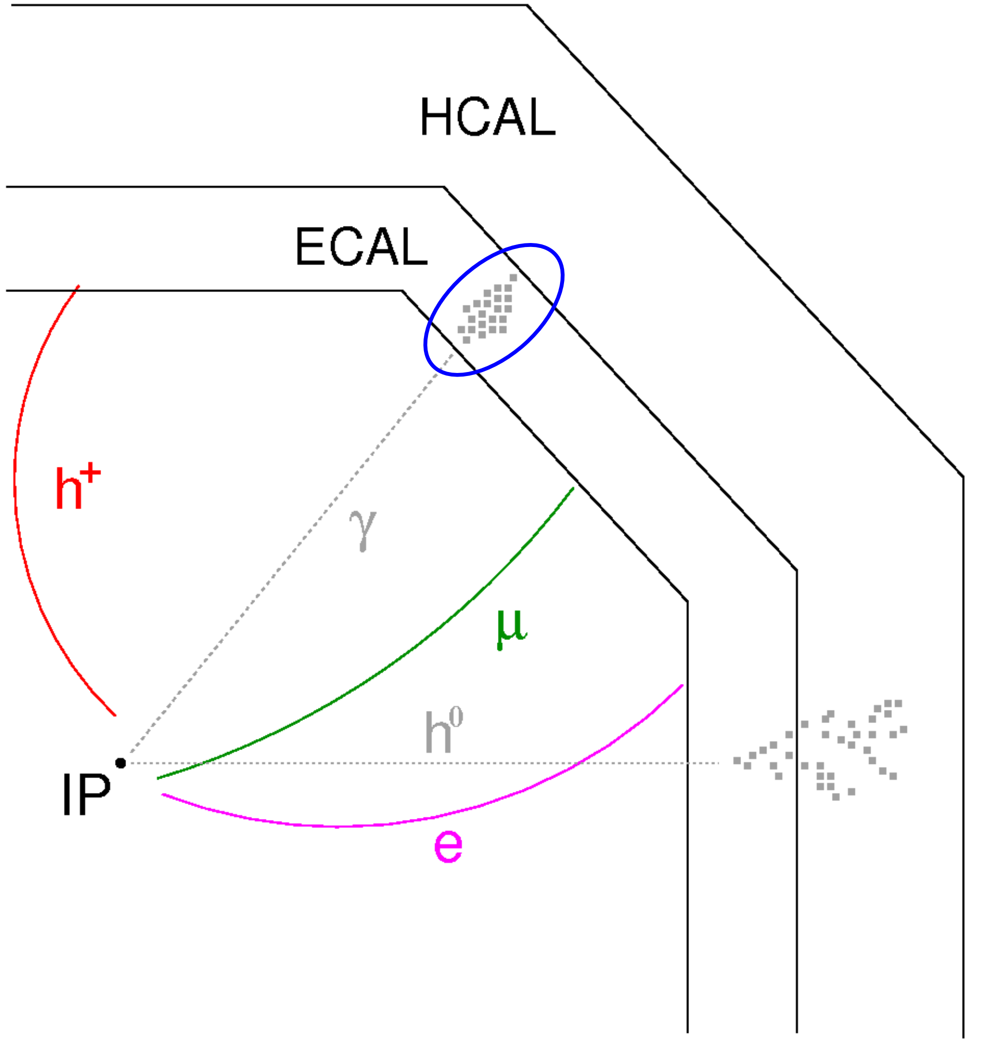
Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)
2. find photon candidates
3. extrapolate tracks into Calorimeter
   - different models, with and w/o energy loss, multiple scattering, ...
   - dedicated Geometry description needed
4. assign MIP stub to track, find muons
5. clustering (ECAL and HCAL)
   - variable, depending on track and photon candidates
   - different algorithms
6. particle ID for $e^{+/-}$, $h^{+/-}$



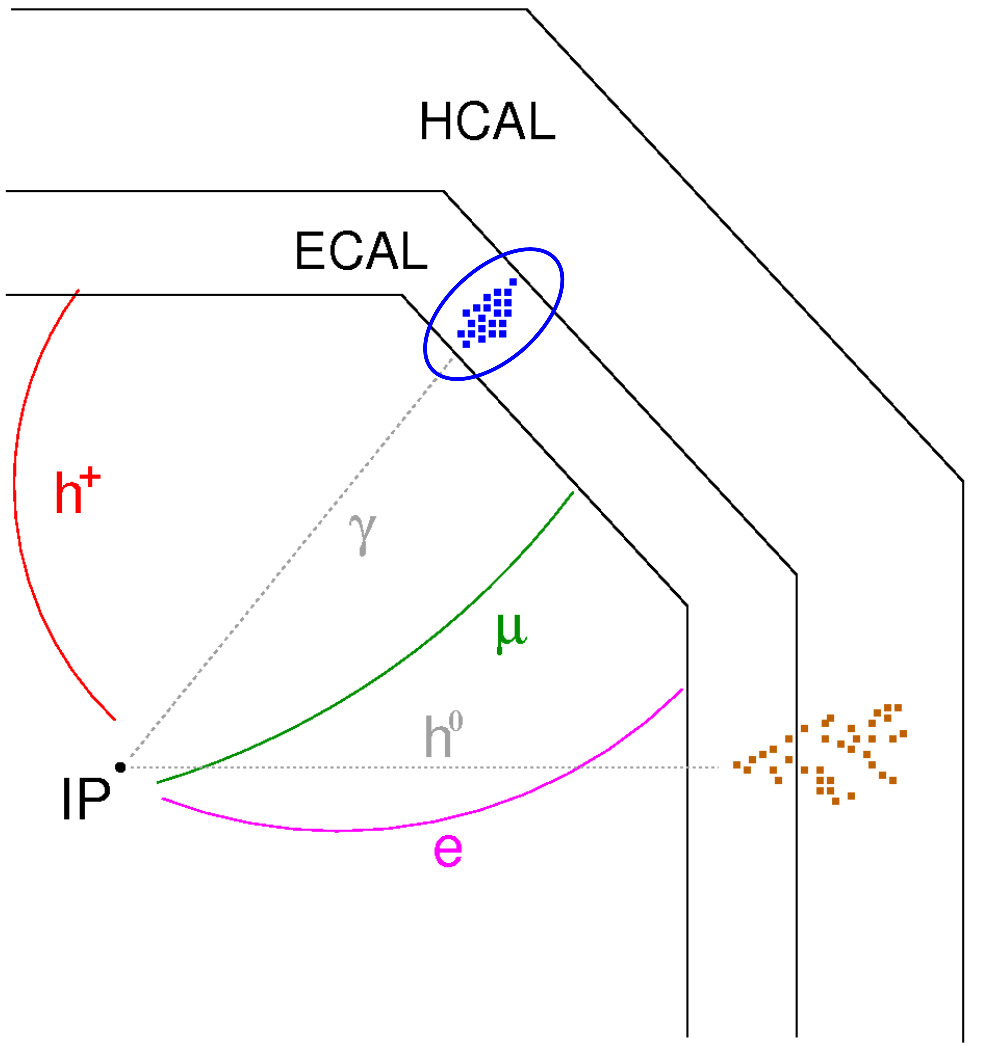
Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)
2. find photon candidates
3. extrapolate tracks into Calorimeter
   - different models, with and w/o energy loss, multiple scattering, ...
   - dedicated Geometry description needed
4. assign MIP stub to track, find muons
5. clustering (ECAL and HCAL)
   - variable, depending on track and photon candidates
   - different algorithms
6. particle ID for $e^{+/-}$, $h^{+/-}$
7. remove 'charged' Calorimeter hits



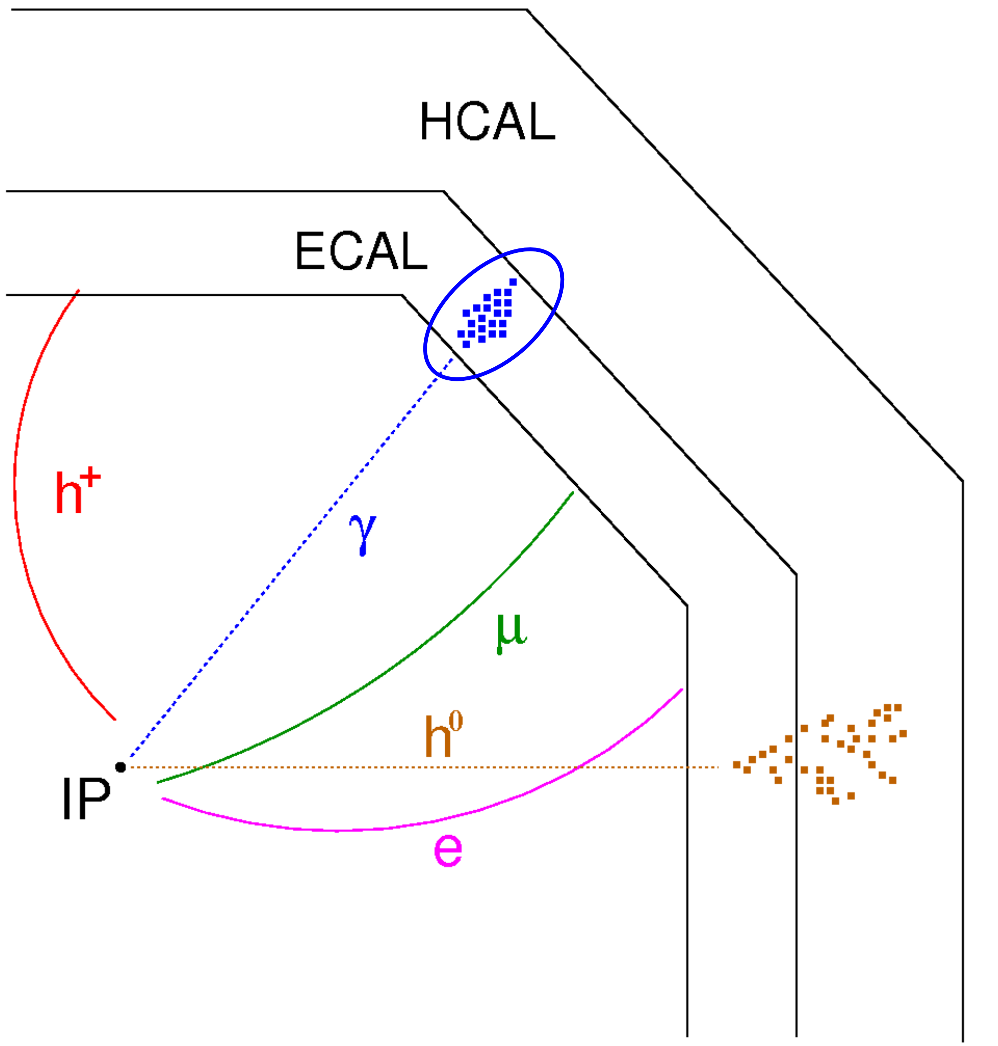
Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)
2. find photon candidates
3. extrapolate tracks into Calorimeter
   - different models, with and w/o energy loss, multiple scattering, ...
   - dedicated Geometry description needed
4. assign MIP stub to track, find muons
5. clustering (ECAL and HCAL)
   - variable, depending on track and photon candidates
   - different algorithms
6. particle ID for $e^{+/-}$, $h^{+/-}$
7. remove 'charged' Calorimeter hits
8. clustering on 'neutral' hits



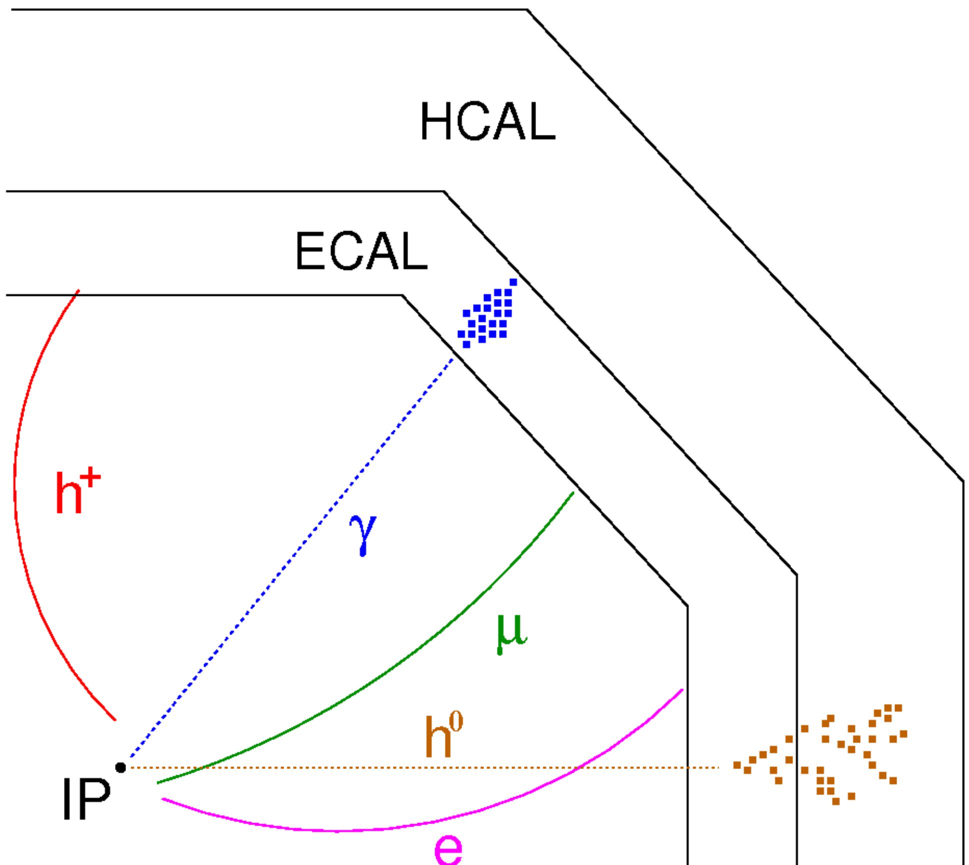
Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)
2. find photon candidates
3. extrapolate tracks into Calorimeter
   - different models, with and w/o energy loss, multiple scattering, ...
   - dedicated Geometry description needed
4. assign MIP stub to track, find muons
5. clustering (ECAL and HCAL)
   - variable, depending on track and photon candidates
   - different algorithms
6. particle ID for $e^{+/-}$, $h^{+/-}$
7. remove 'charged' Calorimeter hits
8. clustering on 'neutral' hits
9. particle ID for photons and h^0



Track-Based Particle Flow Concept

1. tracking (Silicon and TPC)
2. find photon candidates
3. extrapolate tracks into Calorimeter
   - different models, with and w/o energy loss, multiple scattering, ...
   - dedicated Geometry description needed
4. assign MIP stub to track, find muons
5. clustering (ECAL and HCAL)
   - variable, depending on track and photon candidates
   - different algorithms
6. particle ID for $e^{+/-}$, $h^{+/-}$



first version of such a **'track-based'** Particle Flow approach available in MarlinReco

- full software chain (put in tracks and calorimeter hits, get out reconstructed particles)
- more or less modular approach
- first results **only** for $Z^0 \rightarrow uds$ @ 91.2 GeV