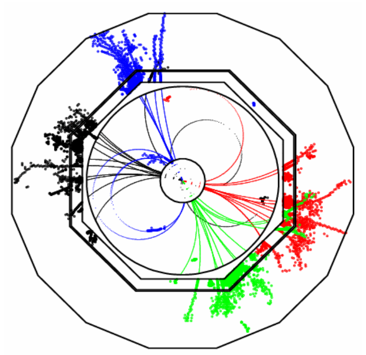


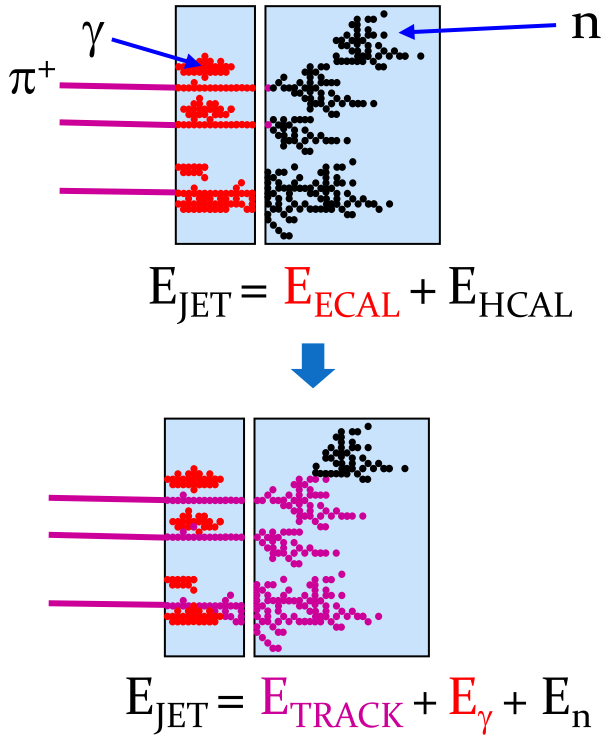
Redesign of Pandora PFA

John Marshall,
University of Cambridge

IWLC2010, Geneva, October 19 2010

Overview

- In a typical jet:
  - 60% of jet energy is in form of charged hadrons
  - 30% is in photons (mainly from $\pi^o \rightarrow \gamma\gamma$)
  - 10% is in neutral hadrons (mainly n and K_L)


γ
π^+
n
$E_{JET} = E_{ECAL} + E_{HCAL}$
$E_{JET} = E_{TRACK} + E_\gamma + E_n$

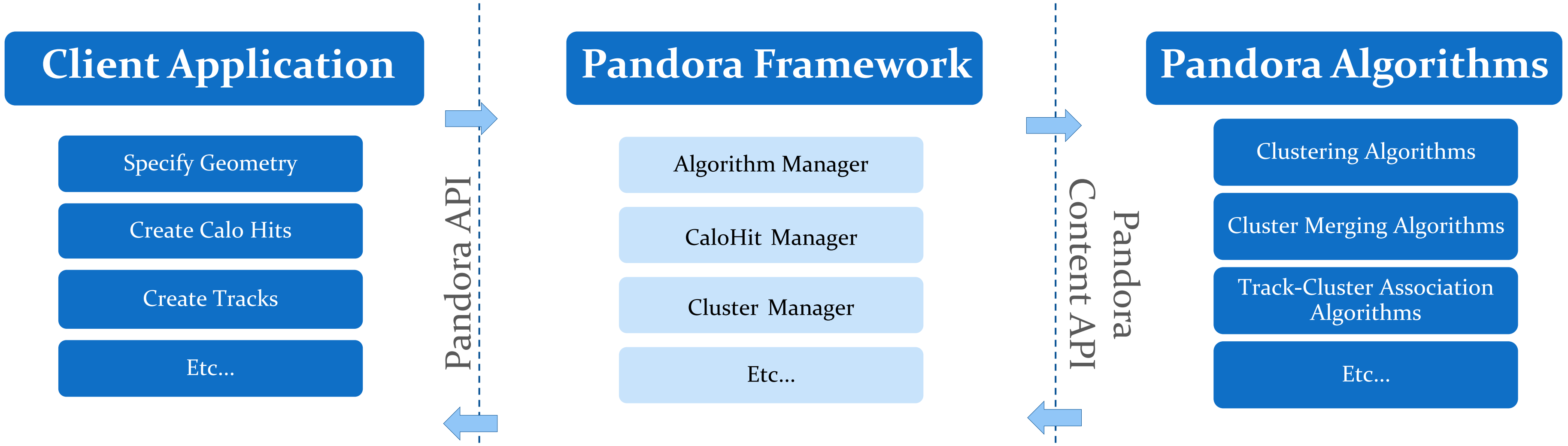
- Particle flow calorimetry aims to improve jet energy resolution by:
  - Measuring charged particles in detector tracker
  - Measuring photon energies in ECAL
  - Measuring only neutral hadron energies in HCAL

- Pandora is the most mature Particle Flow Algorithm, offering the best performance and an approach that is understood and well-documented. However, by 2009 the code had become too difficult to modify or extend.

- During the past year, a new framework for running decoupled particle flow algorithms has been developed and the original Pandora code has been fully reimplemented within the framework.

New Pandora Structure

In the new framework, Pandora is divided into three sections. Communication between these sections is achieved via simple C++ APIs, Application Programming Interfaces, which each provide a “high-level” service:



Isolates specific details of software framework and detector, allowing framework to be dependency-free.

Specifies parameters for calo hits, tracks, etc. so that Pandora objects can be self-describing.

Owns named collections of Pandora objects: calo hits, tracks, clusters and PFOs.

Can perform memory management, as content can only be provided or accessed via APIs.

Use APIs to access Pandora objects and carry out particle flow reconstruction tasks.

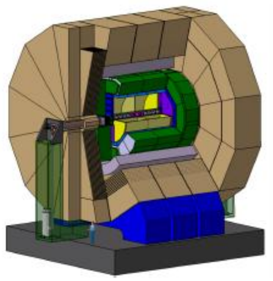
Physics-driven code, with nested structure promoting re-use of code to perform specific tasks.

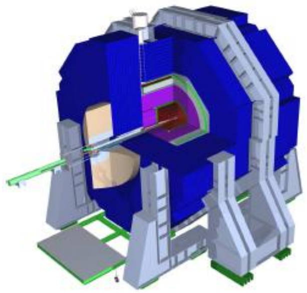
Client Application

- A Pandora client application, written in any software framework, uses the PandoraAPI to supply details of the detector geometry and the hits/tracks/MCParticles in each event.
- Pandora then builds its own self-describing objects to use in the reconstruction.
- Construction of Pandora objects is designed to be simple; client application makes a Parameters class, supplies all the required information and calls Create API.
- Required information is self-describing and physics-based, e.g. for a track:

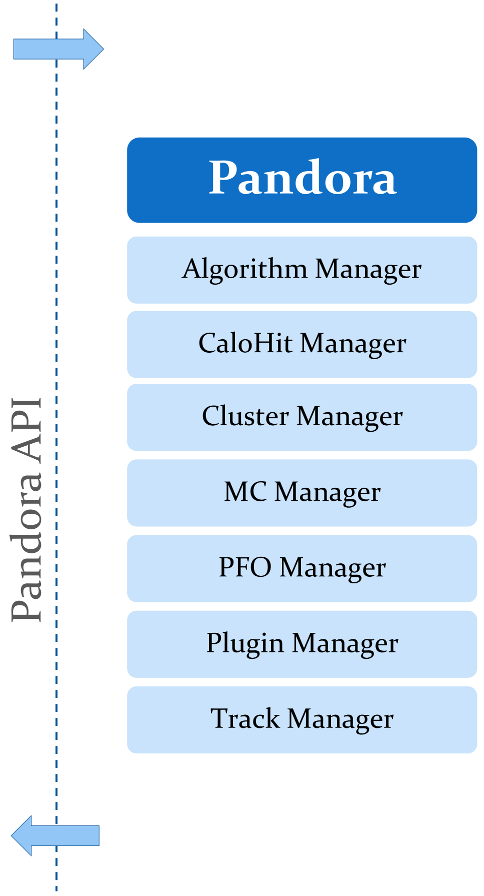
  d0, z0, track state at start, track state at ECal, etc.

- After object creation is complete, client application calls ProcessEvent API. When the thread is returned, client application can call GetParticleFlowObjects API.


ILD/Marlin

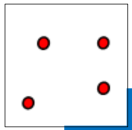

SiD/org.lcsim

+LAr TPC reconstruction
+LHC experiments



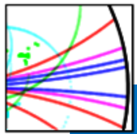
Applications all use same Pandora framework & algorithms

Pandora Objects



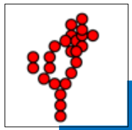
Calo Hit

- Position + normal vectors
- Calorimeter cell size
- Absorber material in front of cell
- Time of first energy deposition
- Calibrated energy (mip equivalent, EM, Had)
- Layer + pseudolayer
- Hit type + detector region
- Density weight
- Surrounding energy
- IsDigital, IsIsolated + IsPossibleMip flags
- Associated MC particle
- Associated user object



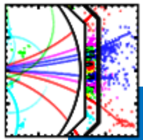
Track

- 2D impact parameters
- Momentum at d.c.a
- Particle mass
- Charge sign
- Start track state
- End track state
- ECal track state
- ReachesECal flag
- List of track state projections to calorimeter surfaces
- Associated cluster
- Associated MC particle
- Associated user object
- PFO formation flag
- “Clusterless” PFO formation flag



Cluster

- List of constituent calo hits, ordered by pseudolayer
- Mip fraction
- EM energy measure
- Had energy measure
- Initial direction
- Current direction
- Result of linear fit to all hits in cluster
- Energy-weighted centroid
- ShowerStart layer
- Shower profile properties
- List of associated tracks



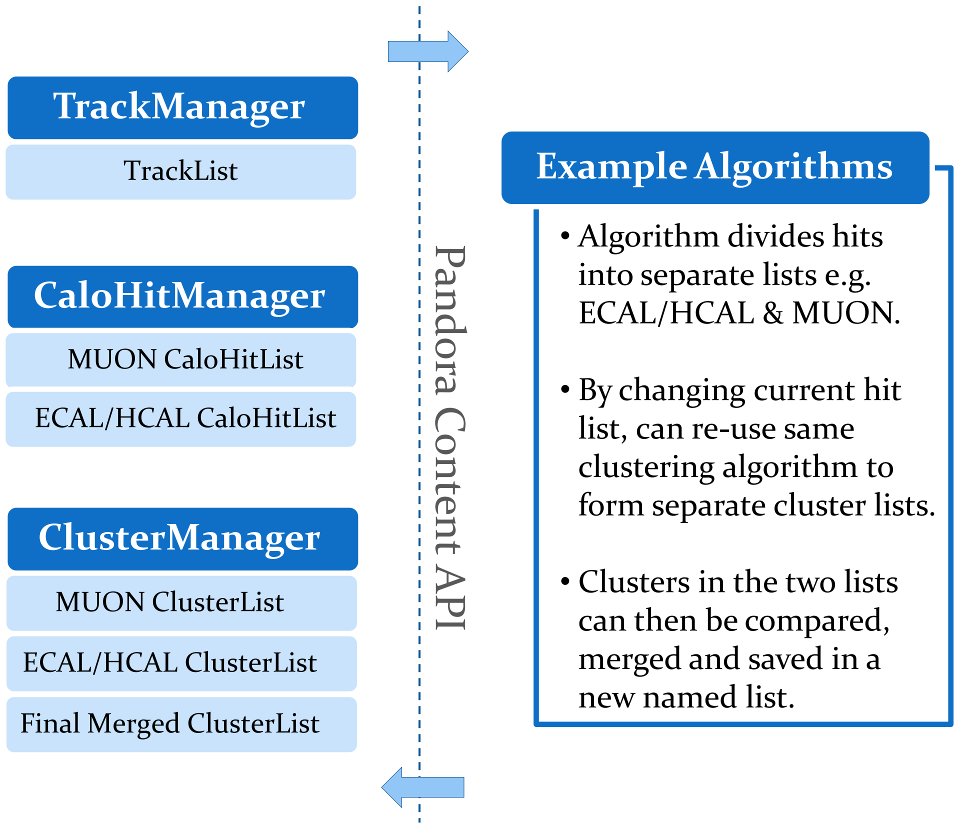
Particle Flow Object

- PDG Code
- Charge
- Mass
- Energy
- Momentum
- List of tracks
- List of clusters

The Pandora objects provide a mixture of properties specified by the user and value-added properties. They are all simple and well defined physics quantities for use in the particle flow algorithms.

Pandora Framework

- The Pandora Managers store named lists of objects. These can be accessed by Pandora algorithms, which perform the reconstruction.
- Algorithms interact with the Managers in a controlled way, via PandoraContentAPI, so the Managers can perform the memory management.
- At any instant, each Manager designates a particular list as the “current list”, which can be requested by an algorithm.
- By manipulating the current lists, parent algorithms can control the scope and behaviour of nested daughter algorithms.
- Algorithms can use the PandoraContentAPI to modify lists and/or save new lists; all the book-keeping is performed by the Managers.



- Algorithm divides hits into separate lists e.g. ECAL/HCAL & MUON.
- By changing current hit list, can re-use same clustering algorithm to form separate cluster lists.
- Clusters in the two lists can then be compared, merged and saved in a new named list.

Algorithms can use the APIs without worrying about how the managers work – separation of physics and C++ memory management!

Pandora Algorithms

- In the new Pandora framework, the algorithms perform the actual particle flow reconstruction. The algorithms should contain almost exclusively physics-driven code, calling high-level APIs to perform the following operations:
  - Create new clusters and particle flow objects
  - Modify clusters, by adding hits, merging or deleting
  - Access the current lists of Pandora objects
  - Save new lists of clusters, calo hits or tracks
  - Run a daughter algorithm, etc…

- Pandora algorithms should contain just the kernel of a well-defined operation.

- Algorithms can be "blind" to their external environment, allowing for efficient re-use of code when the current lists are changed.

- Helper functions are provided to perform tasks useful to multiple algorithms, such as performing a linear fit to layers of a cluster.

Cluster Merging

- Asks for current cluster list.
- Loop over list to identify associated clusters e.g. nearby clusters that point towards each other.
- Ask to merge candidate clusters.

Cluster-Track Association

- Asks for current track and cluster lists.
- Loop over lists to identify cluster with most compatible position and direction for each track.
- Ask to form track-cluster association.

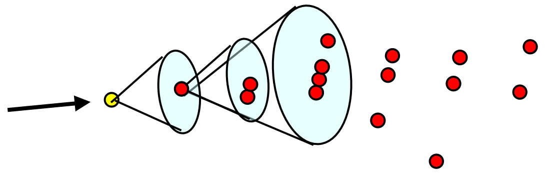
Example algorithms

Example Algorithm: Clustering

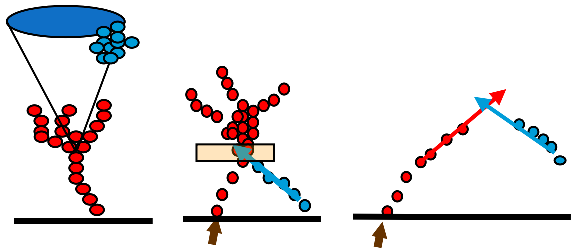
To reproduce original Pandora clustering algorithm in new framework, the requirements are:

i. A parent algorithm to control operations,
ii. A cluster formation algorithm,
iii. Topological association algorithms (optional).

1. Parent algorithm asks to run a clustering algorithm; cluster manager then creates a new temporary cluster list, associated with the parent algorithm, sets this as "current", and allows new clusters to be formed.



2. Daughter clustering algorithm gets current hit and track lists, populates temporary cluster list and returns control to parent algorithm.



3. Parent algorithm calls topological association algorithms, which cannot form new clusters, but can modify or merge existing clusters.

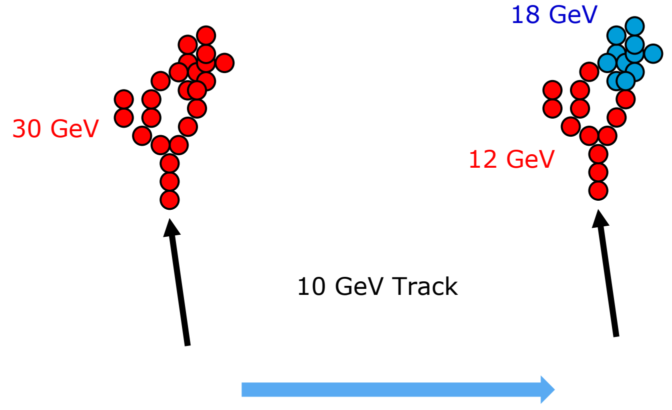
4. Parent algorithm can save the temporary clusters as a new named cluster list. Can set new list as current list for future algorithms, if desired. Any remaining temporary clusters will be tidied automatically.

Example Algorithm: Reclustering

An important part of the original Pandora reconstruction is "reclustering"; attempting to redistribute hits between clusters in order to improve consistency between cluster energies and associated track momenta.

Parent reclustering algorithm operations:

1. Identify inconsistent pairing of track and cluster(s) and ask to recluster these.
   - Relevant clusters moved to new temporary cluster list. Current hit/track lists changed.
2. Ask to run a clustering algorithm.
   - Creates another uniquely named temporary cluster list, filled by daughter clustering algorithm.
3. Calculate figure of merit for consistency of track and new cluster(s).
4. Repeat stages 2. and 3. as required.
   - Can re-use original clustering algorithm, with different parameters, or try entirely new algorithms.
5. Choose most appropriate cluster(s).
   - All lists will be reorganised and tidied accordingly.



Change clustering parameters and/or clustering algorithm until cluster splits and get sensible track-cluster match

Algorithm Configuration

- The Pandora framework is designed to make algorithm tuning painless. Algorithms can be swapped in/out without recompiling and there are no hard-coded parameters.
- Algorithms are configured via xml; a natural way to configure nested structures:
  - Ideal for quickly experimenting with running new algorithms.
  - Easy to mix "real" and "cheating" algorithms.
  - The process of reclustering is reduced to the following simple configuration:

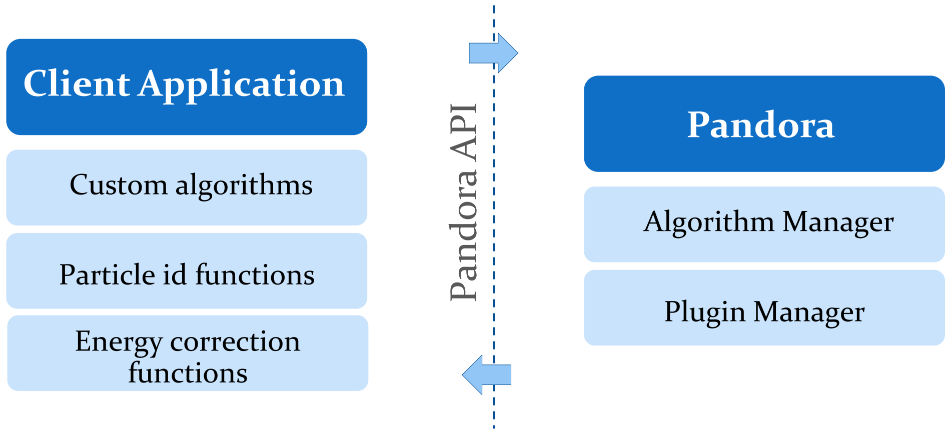
```
<algorithm type = "Reclustering”>
    <!-- List of daughter clustering algorithms -->
    <clusteringAlgorithms>
        <algorithm type = "ClusteringType1”> ClusteringType1 parameters ... </algorithm>
        <algorithm type = "ClusteringType2”> ClusteringType2 parameters ... </algorithm>
        <algorithm type = "ClusteringTypeN”> ClusteringTypeN parameters ... </algorithm>
    </clusteringAlgorithms>

    <!-- Other parent reclustering algorithm properties -->
</algorithm>
```

- Every algorithm has a ReadSettings method, called at startup. In this method, algorithm parameters are assigned default values and the xml file is parsed to see if values have been overridden.
- Algorithms within the top-level xml tags are automatically called (in order) for each event. These algorithms can, in turn, call their own daughter algorithms.

Customising Pandora

- To make it easy for people to get involved and try out new ideas, Pandora users can register and use their own (existing) content, including:
  - Particle identification functions,
  - Hadronic and electromagnetic energy correction functions,
  - Custom particle flow algorithms, allowing for a completely different reconstruction.

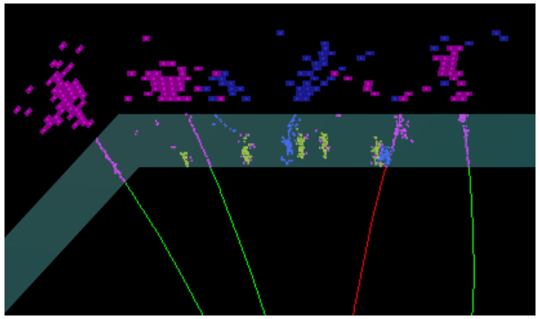


- Pandora API is used to pass addresses of helper functions and algorithm “factories” to managers.
- Gives Pandora the ability to call the helper functions and to create and run instances of the user algorithms.

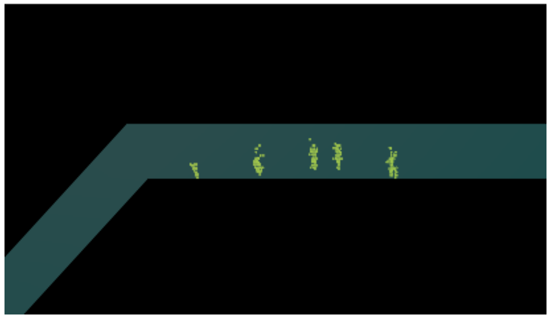
- The algorithms and helper functions compiled as part of the Pandora library should have no dependencies and be (subject to steering parameters) detector independent. This need not apply to custom content.
- This provides a new opportunity to bring together many independent analyses within Pandora…

Including External Content

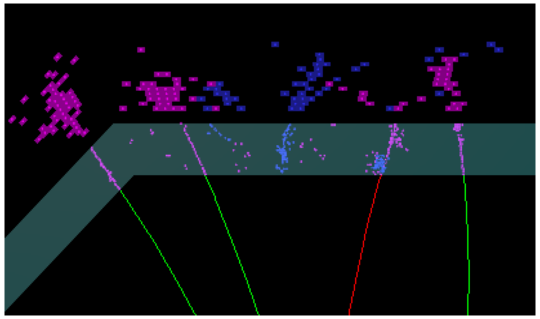
- The ability to create custom algorithms, compiled as part of the client application, means it is trivial to write simple wrapper algorithms for external content.
- These algorithms can bring results from external packages right into the Pandora reconstruction. For example, can very simply (~50 lines of code) use output from GARLIC to replace photon identification.


Final Pandora reconstruction

=


GARLIC photon clusters

+


Standard Pandora reconstruction for remaining hits

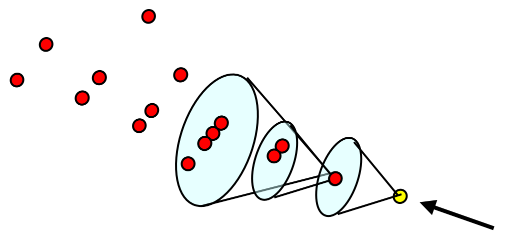
- Example procedure for adding photon clusters identified by an external package:

  i. Run GARLIC as normal, saving output photon clusters in lcio collection
  ii. Early in Pandora reconstruction, use wrapper algorithm to recreate photon clusters within Pandora.
  iii. Tag clusters as photons and save in a named list, removing hits from subsequent reconstruction.
  iv. Add the photon clusters to back into reconstruction just before PFO construction.

Original Pandora

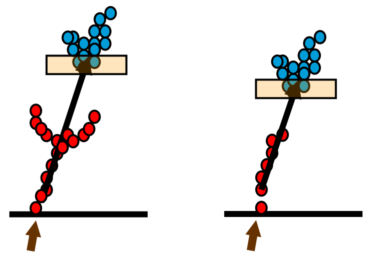
Having created the new framework, reimplementation of the original Pandora code could begin. This required implementation of the following algorithms:

Clustering Algorithm



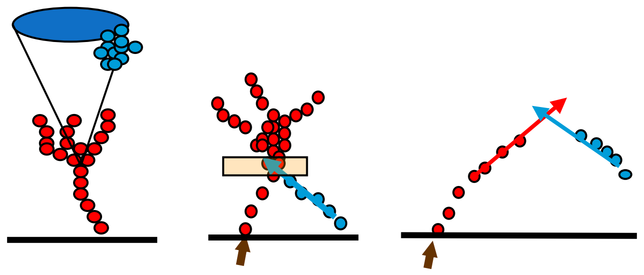
Cone-based forward projective method

Topological Association Algorithms



Proximity

Track segment pointing to shower

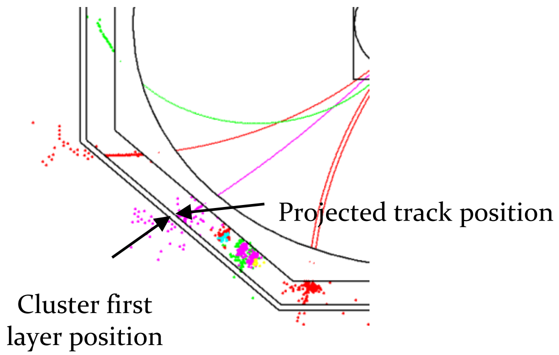


Cone associations

Back-scattered tracks

Looping tracks

Track-cluster Association Algorithms

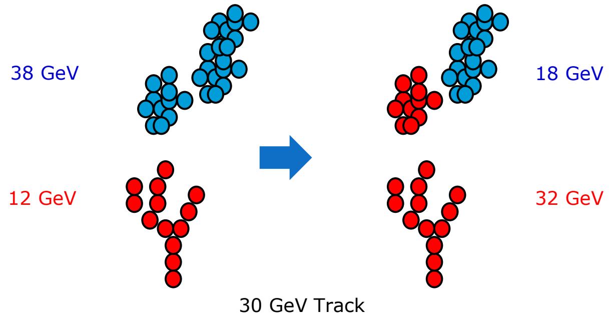


Reclustering Algorithms

Continued

Original Pandora

Continued



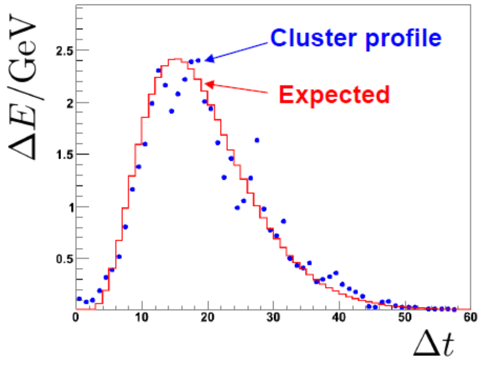
Track-cluster Association Algorithms

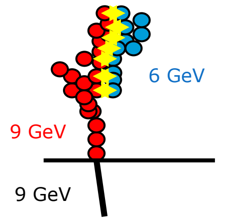
Reclustering Algorithms

Photon Recovery Algorithms

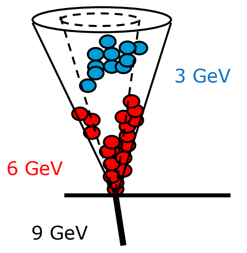
Fragment Removal Algorithms

PFO Construction Algorithms

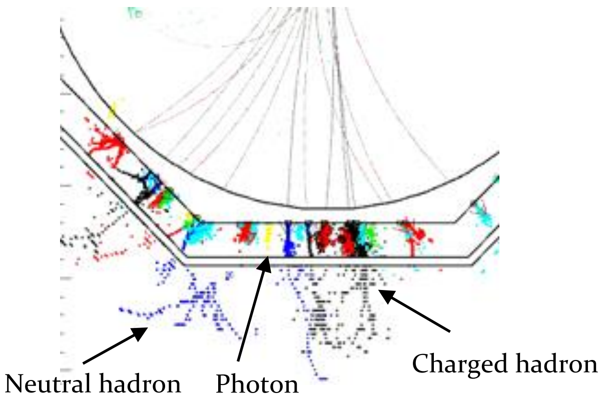




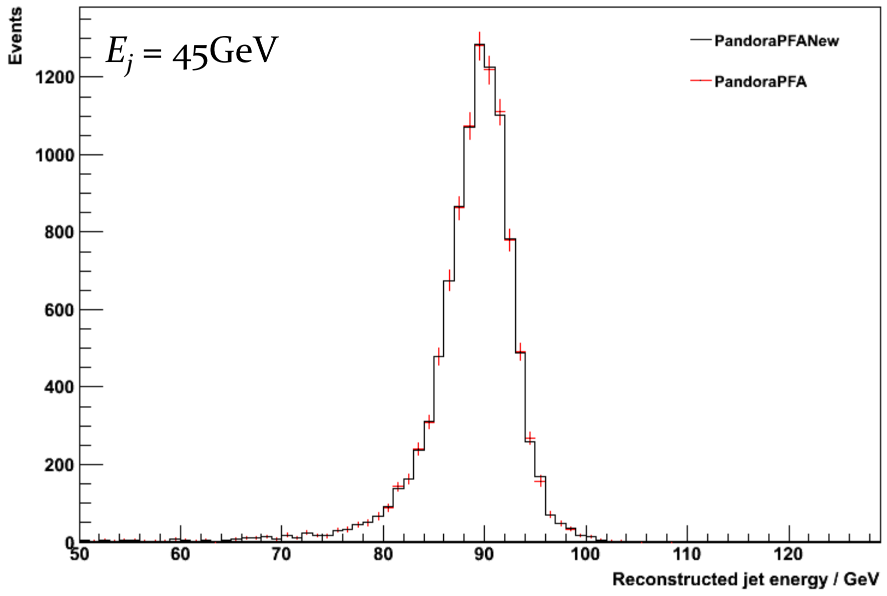
Layers in close contact

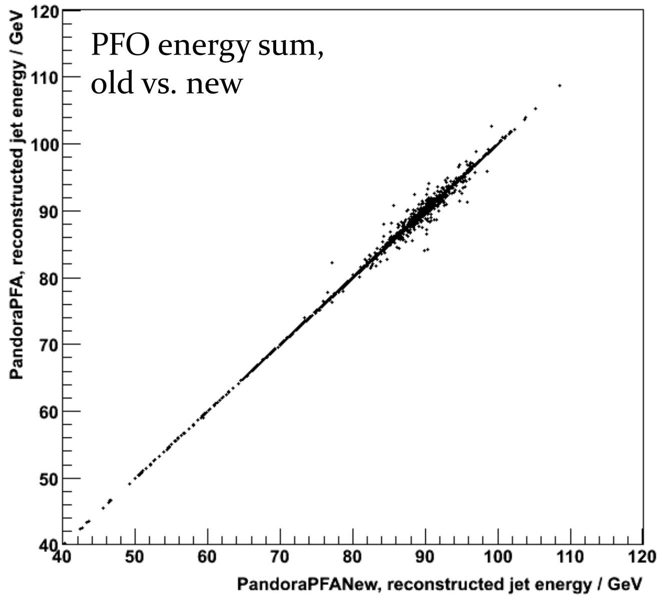


Fraction of energy in cone



Performance at LCWS10

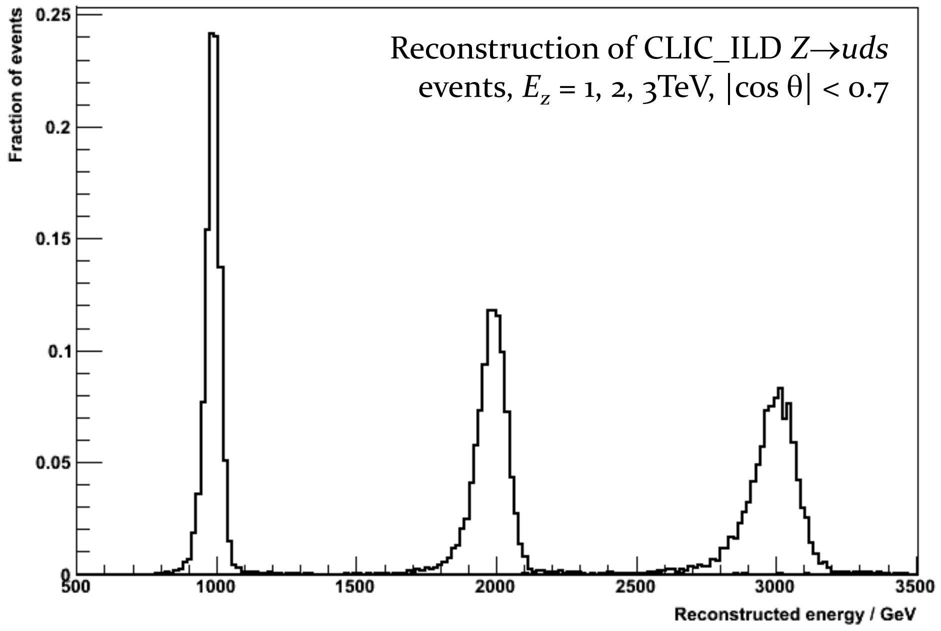




- After reimplementation, the first tests were performed for ILD, using MC samples of approximately 10,000 $Z \rightarrow uds$ generated with the Z decaying at rest with E_Z = 91.2GeV.
- At this energy, all newly implemented code and framework was exercised, without (a strong) need for statistical reclustering and/or leakage corrections (at the time, was still work in progress for new Pandora).
- **Excellent agreement observed, by construction.**

Current Performance

Performance for high jet energies (E_j > 500 GeV) is a very active area of development. A number of important changes have been made recently e.g. “forced” clustering if no suitable reclustering candidates are found.



CLIC_ILD, E_j	45GeV	100GeV	250GeV	500GeV	1TeV	1.5TeV
PandoraPFANew, $rms_{90}(E_j) / E_j$	3.55 ± 0.11	2.93 ± 0.08	2.84 ± 0.07	2.96 ± 0.04	3.19 ± 0.04	3.21 ± 0.05

ILD, E_j	45GeV	100GeV	180GeV	250GeV
PandoraPFANew, $rms_{90}(E_j) / E_j$	3.62 ± 0.05	2.94 ± 0.04	3.10 ± 0.04	3.29 ± 0.04

Summary

- A robust framework for running decoupled particle flow algorithms has been implemented and validated.
- Detector or software-specific issues are resolved in a Pandora client application, allowing for re-use of the same Pandora algorithms for e.g. ILD and SiD.
- Pandora algorithms perform the particle flow reconstruction, using high-level API functions to communicate with the Pandora framework, which will perform memory management operations.

- The new framework allows customisation with user-specific algorithms, particle identification functions and energy correction functions.
- Simple wrapper algorithms allow content from external software packages to be quickly included in the Pandora reconstruction.

- The original version of Pandora has been reimplemented and fully validated within the new framework; its jet energy reconstruction performance continues to impress.
- Since this reimplementation, many improvements have been made to the high energy jet reconstruction and particle identification performance.

If anyone is interested in contributing ideas or content to Pandora, please get in touch.