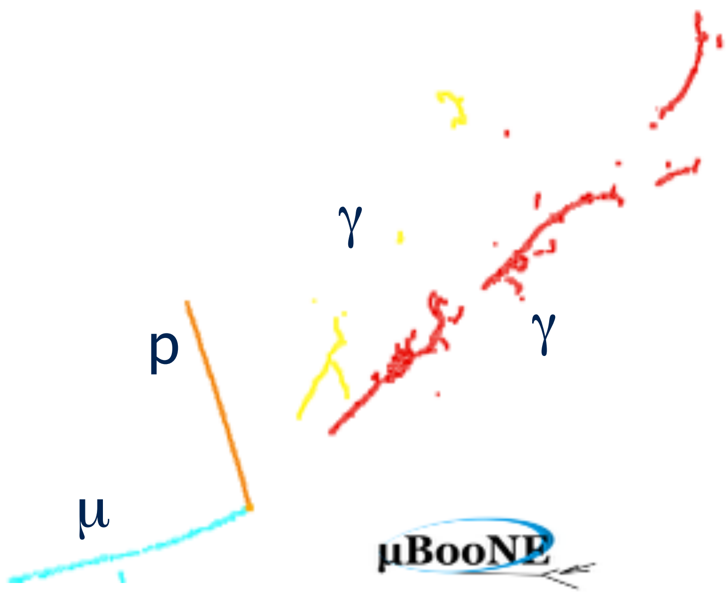
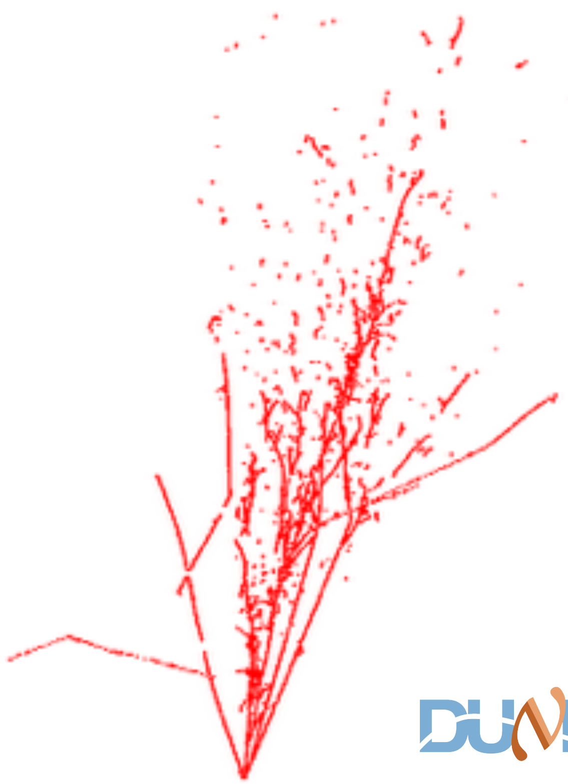
Pandora Talk 4: 2D Reconstruction



A. S. T. Blake for the Pandora Team

MicroBooNE Pandora Workshop

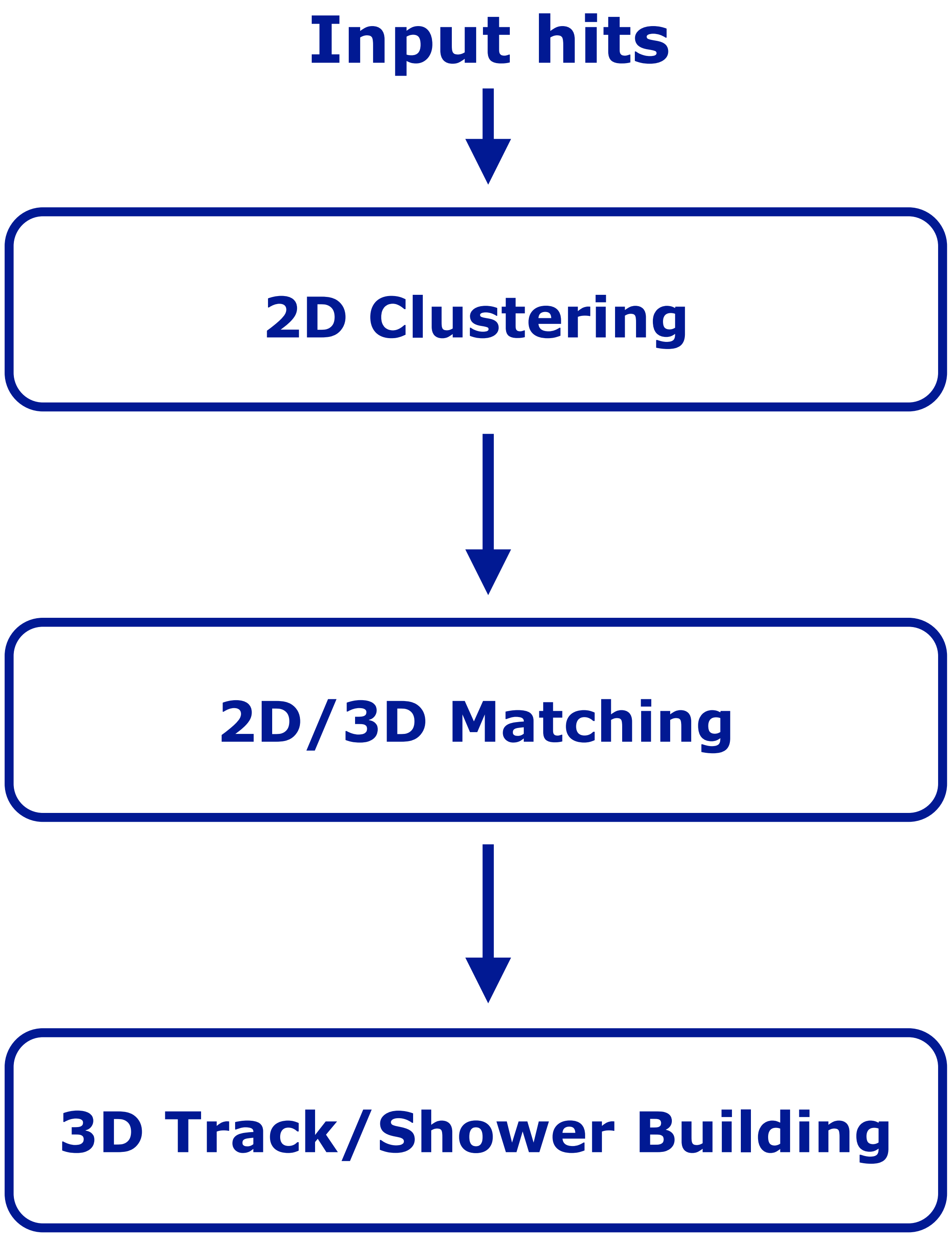
July 11-14th 2016, Cambridge



Overview

- Introduction:
  - ➢ 2D reconstruction within overall context:
    - (a) Overall 2D/3D reconstruction flow.
    - (b) Reconstruction chain for MicroBooNE.
- Overview of 2D pattern recognition algorithms:
  - ➢ Concepts, design patterns, etc.
- Review of 2D pattern recognition algorithms:
  - ➢ Track-based algorithms.
  - ➢ Shower-based algorithms.
  - ➢ "Mop-up" algorithms.
- Summary

2D/3D Reconstruction Flow



2D/3D reconstruction strategy:

1) Build 2D clusters by making topological associations between input reconstruction hits.

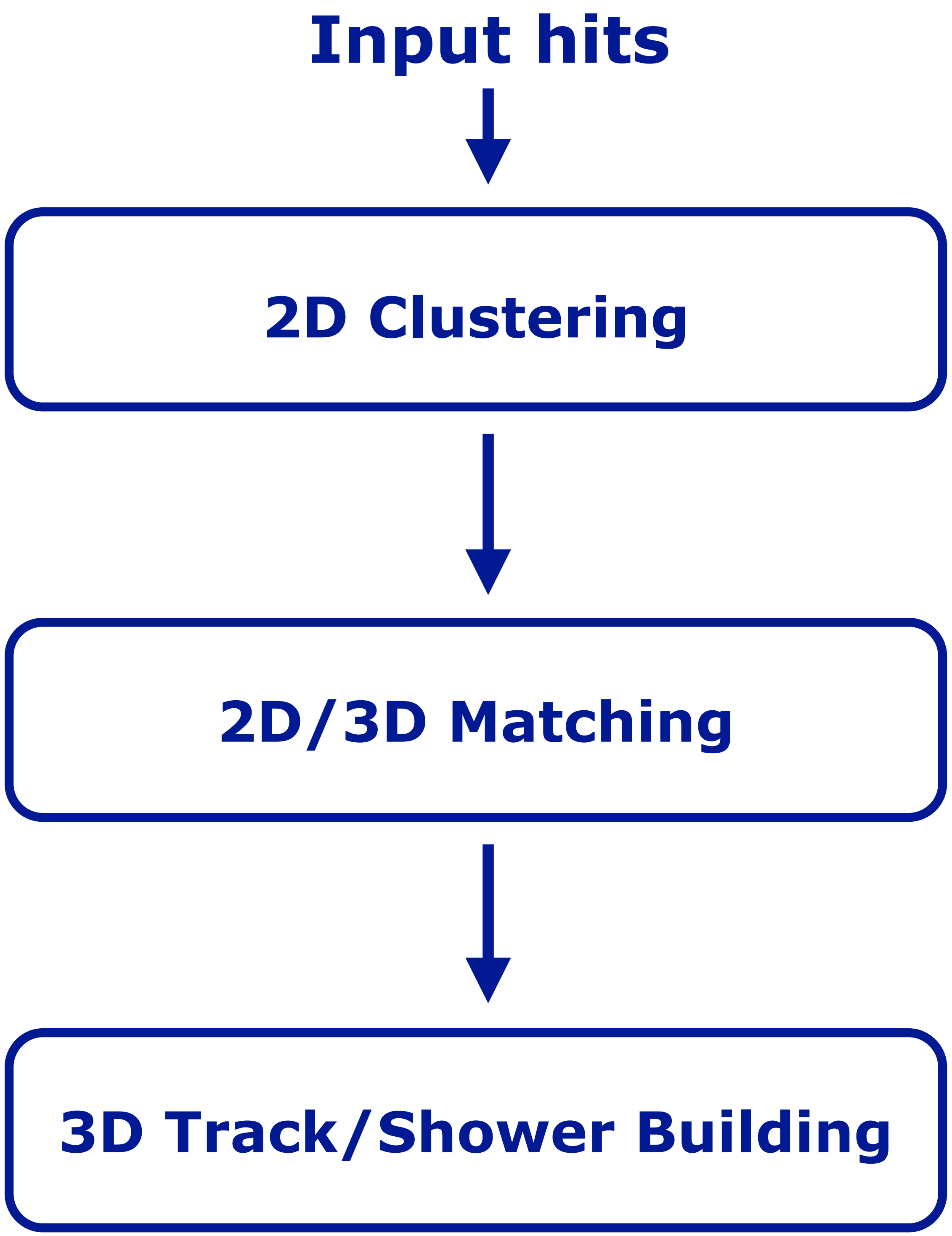
2) Perform 2D/3D matching using iterative approach:

   a) Match 2D clusters between two or three views and form 3D particles.

   b) Also use 3D information to refine 2D clustering.

      [Note: Another key part of the 2D reconstruction!]

3) Generate 3D space points and reconstruct particle hierarchy.



- Note: this flow of reconstruction reflects how we typically analyse events by eye:
  1) Identify patterns in each 2D view first.
  2) Then match up patterns between views (and perhaps correct 2D interpretations).
- Therefore, (I think…) this is a good approach to pattern recognition.
  - ➢ However, still need to write the algorithms (which is difficult…)

2D Reconstruction

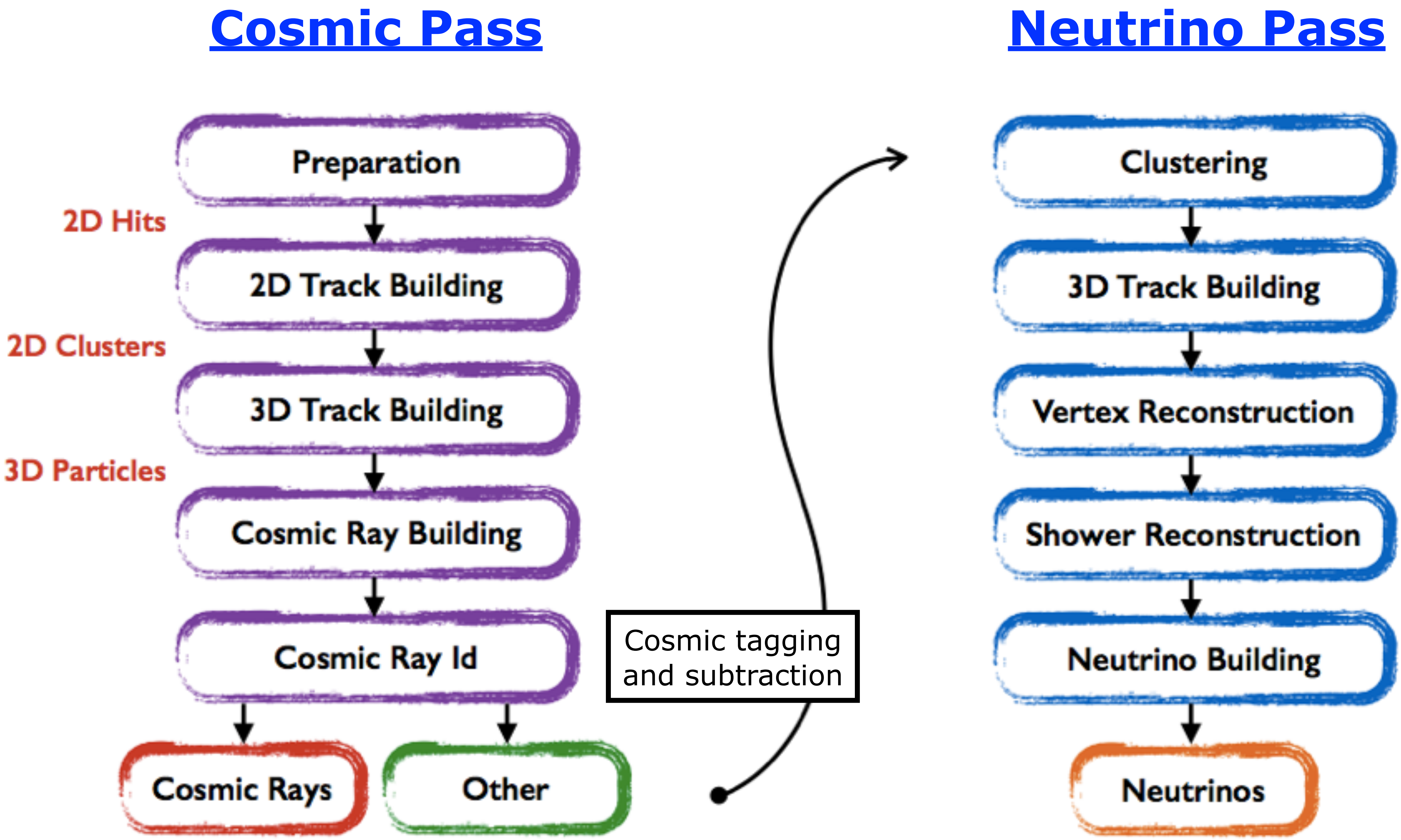
- Purpose of this talk is to present an overview of the 2D elements of the Pandora reconstruction.

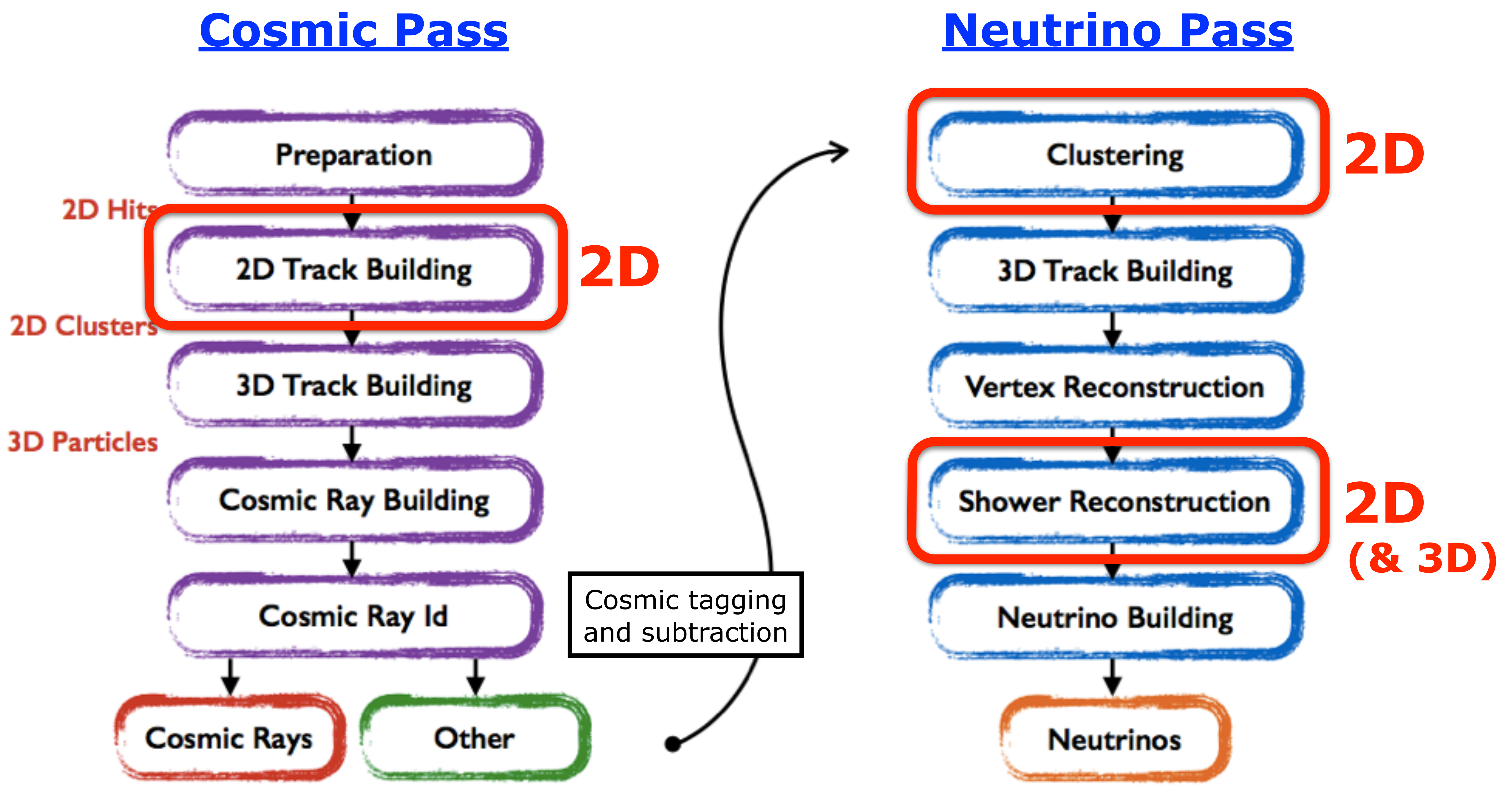
- In practice, the MicroBooNE reconstruction contains several different 2D reconstruction problems:
  - ➢ Track-finding: Search for continuous lines of hits.
  - ➢ Shower-finding: Search for broad clusters of hits.
  - ➢ Cosmic pass: Reconstruct single long tracks from end to end.
  - ➢ Neutrino pass: Reconstruct multiple tracks emerging from a vertex.

- Each of these reconstruction problems is different, and requires a (slightly or very) different approach.
  - ➢ In practice, we combine a common set of multi-purpose algorithms with a number of dedicated algorithms.

- Note: our 2D track reconstruction is probably more mature than our 2D shower reconstruction at the present time.

MicroBooNE Reconstruction





Cosmic Pass
Neutrino Pass
Preparation
2D Hits
2D Track Building
2D
2D Clusters
3D Track Building
3D Particles
Cosmic Ray Building
Cosmic Ray Id
Cosmic Rays
Other
Cosmic tagging and subtraction
Clustering
2D
3D Track Building
Vertex Reconstruction
Shower Reconstruction
2D
(& 3D)
Neutrino Building
Neutrinos

2D Pattern Recognition

- Pandora 2D reconstruction exploits multi-algorithm approach to pattern recognition (as you have already heard…)
  - ➢ Gradually build up event using many focused algorithms, each with specific purpose.
  - ➢ Most algorithms are iterative/recursive.
- In general, 2D algorithms fall into two categories:
  - ➢ Cluster merging: associate and merge hits and clusters, following well-defined sets of rules.
    - Try to be conservative and avoid making mistakes.
  - ➢ Cluster splitting: split up clusters in a controlled manner, correcting well-defined types of over-clustering.
- 2D algorithms are steered by topological information:
  - ➢ e.g. spatial proximity, directional pointing, intersections … (note: these are also typical features that we look for by eye).
  - ➢ No use of calorimetry information (yet).

Design Patterns

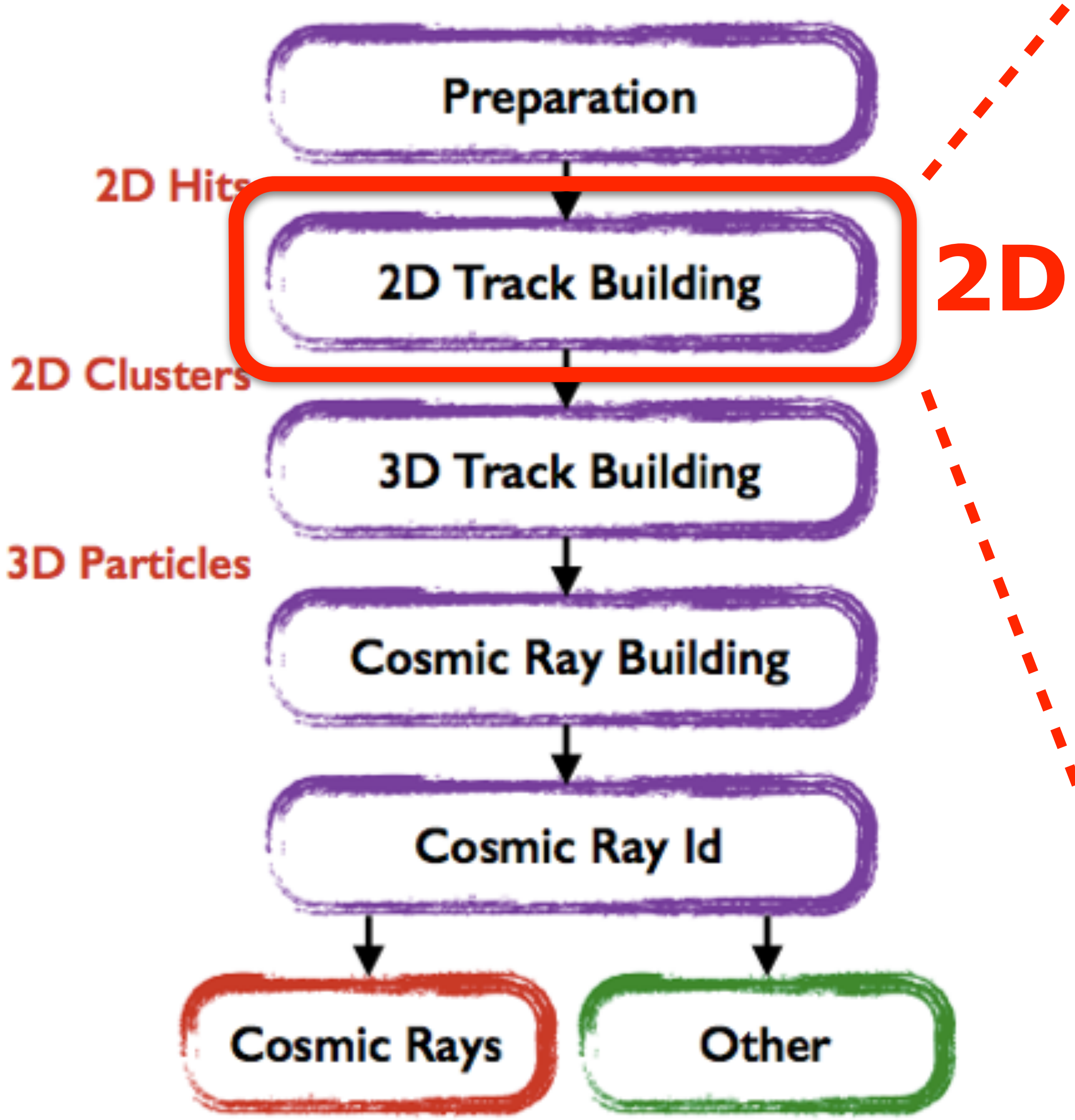
- Many 2D algorithms inherit from common "design patterns", which provide templates for associating and merging clusters.

- Design patterns are typically implemented as follows:

1. Compare every cluster with every other cluster.
   - ➢ Commonly use k-d trees to compare clusters efficiently.
2. Decide which pairs of clusters are "associated".
   - ➢ Inherited algorithms just have to fill in a set of rules. (either implement an 'IsAssociated' method, or fill an 'Association' information block).
3. By analysing associations, decide which clusters to merge. Several possible ways to merge clusters:
   - ➢ Merge together ALL associated clusters; or only consider UNIQUE associations, or choose the BEST associations; or define SEEDS and then follow chains of associations.
   - ➢ Design patterns mainly differ in their merging procedures.

Cosmic Pass

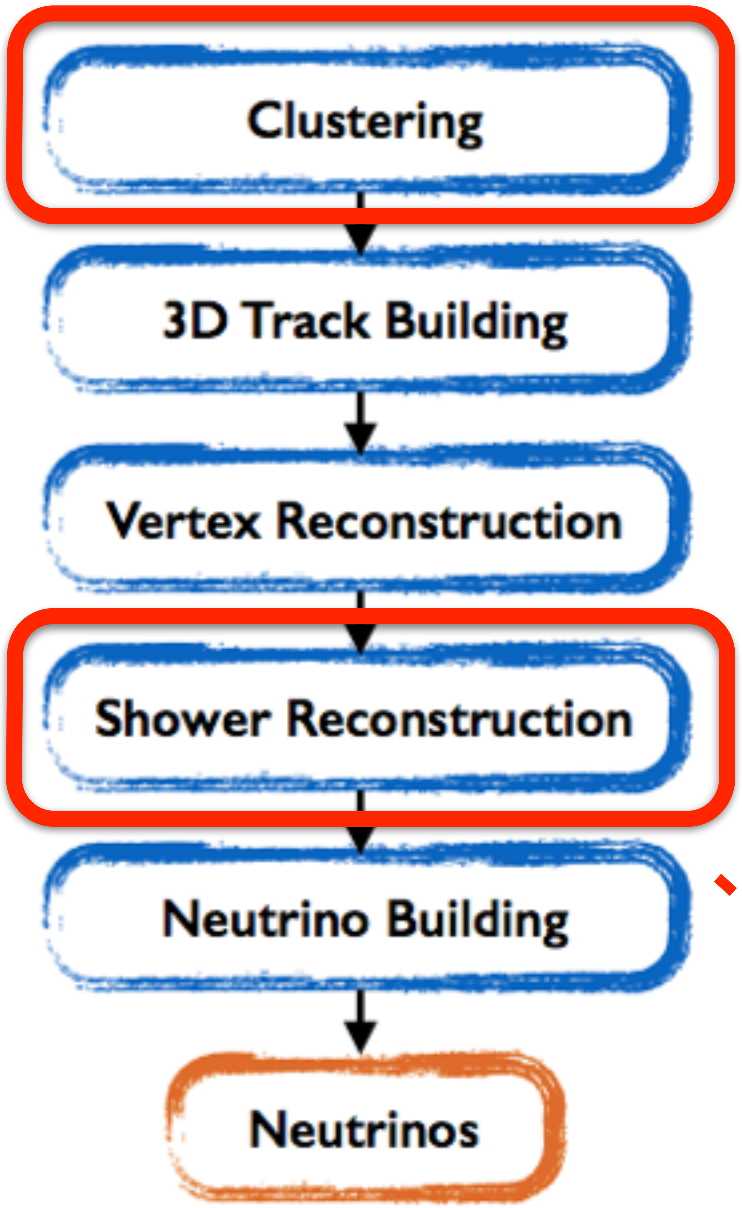
Cosmic Pass



(16 algorithms)

TrackClusterCreation
LayerSplitting
KinkSplitting
TransverseAssociation
LongitudinalAssociation
TransverseExtension
LongitudinalExtension
CrossGapsAssociation
CrossGapsExtension
BranchSplitting
DeltaRaySplitting
CrossedTrackSplitting
CosmicRaySplitting
CosmicRayExtension
DeltaRayExtension
TrackConsolidation

Neutrino Pass



2D

2D (& 3D)

(12 algorithms)

- TrackClusterCreation
- LayerSplitting
- LongitudinalAssociation
- TransverseAssociation
- LongitudinalExtension
- TransverseExtension
- CrossGapsAssociation
- CrossGapsExtension
- OvershootSplitting
- BranchSplitting
- KinkSplitting
- TrackConsolidation

(2 algorithms)

- ClusterCharacterisation
- ShowerGrowing

2D Track Reconstruction

Cosmic Pass

TrackClusterCreation
LayerSplitting
KinkSplitting
TransverseAssociation
LongitudinalAssociation
TransverseExtension
LongitudinalExtension
CrossGapsAssociation
CrossGapsExtension
BranchSplitting
DeltaRaySplitting
CrossedTrackSplitting
CosmicRaySplitting
CosmicRayExtension
DeltaRayExtension
TrackConsolidation

Neutrino Pass

TrackClusterCreation
LayerSplitting
LongitudinalAssociation
TransverseAssociation
LongitudinalExtension
TransverseExtension
CrossGapsAssociation
CrossGapsExtension
OvershootSplitting
BranchSplitting
KinkSplitting
TrackConsolidation

Common algorithms
Cosmic-only algorithms
Neutrino-only algorithms

2D Track Reconstruction: Cosmic Pass

Cosmic Pass

TrackClusterCreation
LayerSplitting
KinkSplitting
TransverseAssociation
LongitudinalAssociation
TransverseExtension
LongitudinalExtension
CrossGapsAssociation
CrossGapsExtension
BranchSplitting
DeltaRaySplitting
CrossedTrackSplitting
CosmicRaySplitting
CosmicRayExtension
DeltaRayExtension
TrackConsolidation

1. Form 2D proto-clusters by merging together lines of contiguous hits.
   - These provide building blocks for subsequent track reconstruction.
2. Split proto-clusters at "corners".
3. Run "growing" algorithms, which make end-to-end joins between proto-clusters.
4. Jump over detector gaps.
5. Split and splice tracks which have mistakenly followed delta rays etc.
6. Extend tracks as far as possible in each direction.
   - Also extend delta-ray showers.
7. Pick up all hits along length of track.

2D Track Reconstruction

Neutrino Pass

TrackClusterCreation
LayerSplitting
LongitudinalAssociation
TransverseAssociation
LongitudinalExtension
TransverseExtension
CrossGapsAssociation
CrossGapsExtension
OvershootSplitting
BranchSplitting
KinkSplitting
TrackConsolidation

1. Form 2D proto-clusters by merging together lines of contiguous hits.
   - These provide building blocks for subsequent track reconstruction.
2. Split proto-clusters at "corners".
3. Run "growing" algorithms, which make end-to-end joins between proto-clusters.
4. Jump over detector gaps.
5. Split and splice tracks which mistakenly tracked through a kink or vertex, or followed delta rays etc.
6. Pick up all hits along length of track.

2D Track Reconstruction: Design Patterns

Cosmic Pass

TrackClusterCreation
LayerSplitting
KinkSplitting
TransverseAssociation
LongitudinalAssociation
TransverseExtension
LongitudinalExtension
CrossGapsAssociation
CrossGapsExtension
BranchSplitting
DeltaRaySplitting
CrossedTrackSplitting
CosmicRaySplitting
CosmicRayExtension
DeltaRayExtension
TrackConsolidation

Can see presence of design patterns in these algorithms:

"Creation"

"Association"

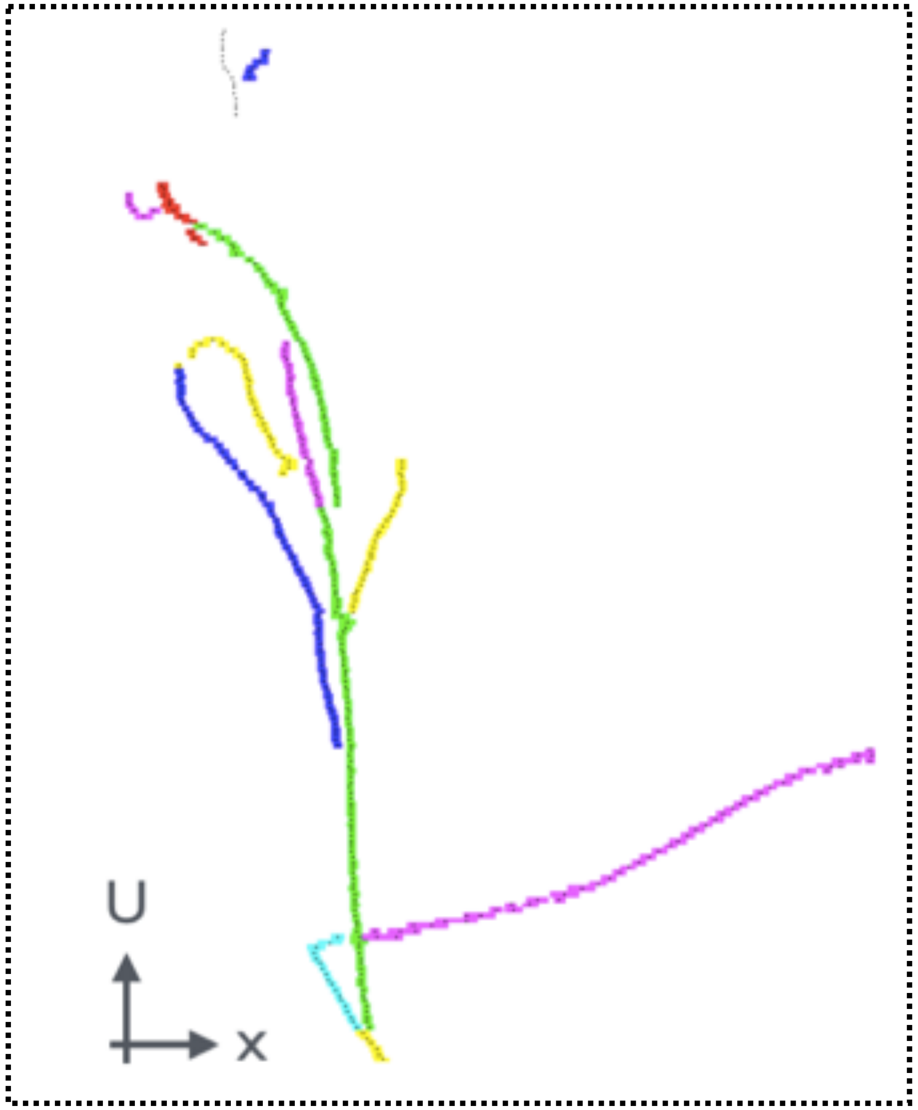
"Extension"

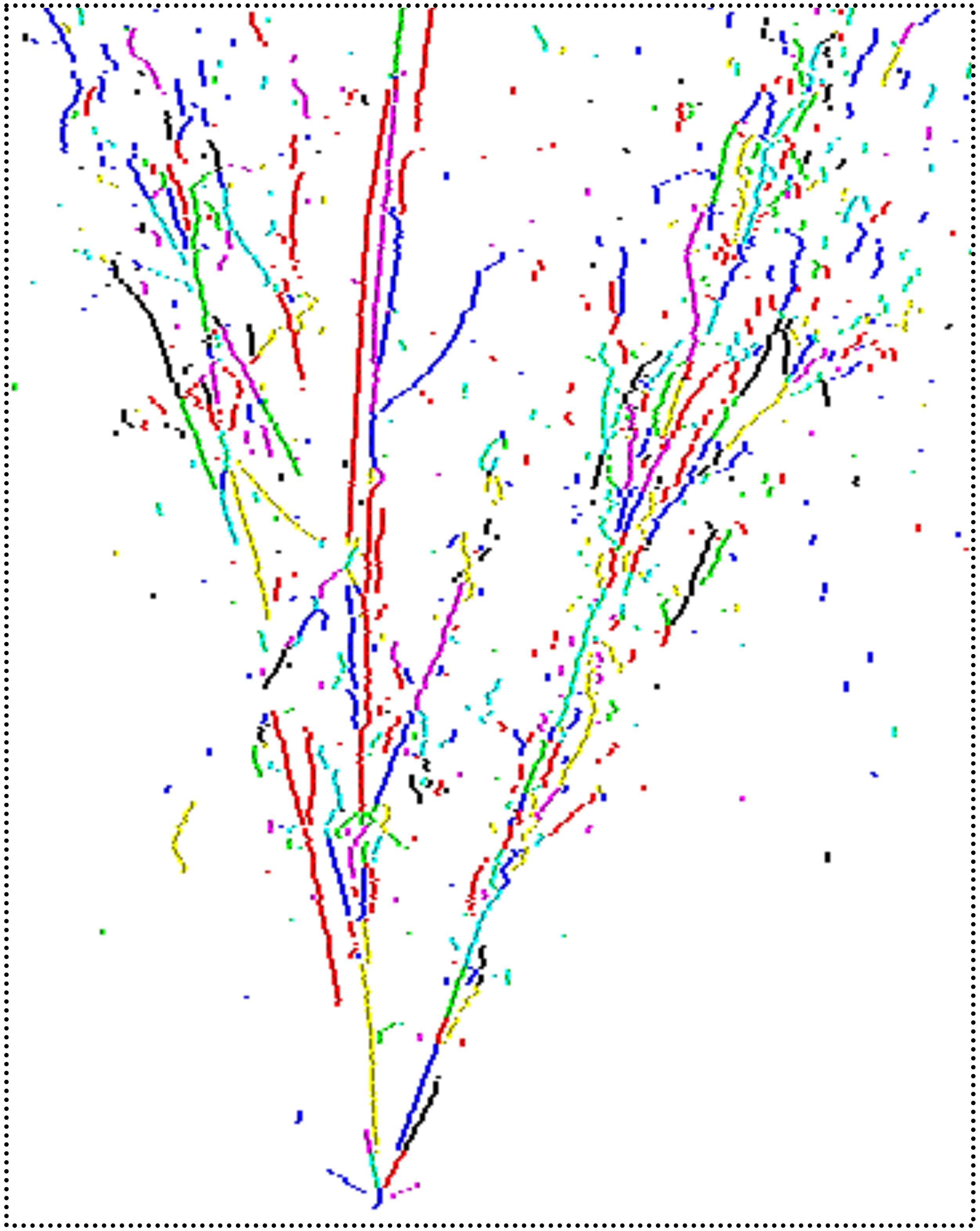
"Splitting"

"Consolidation"

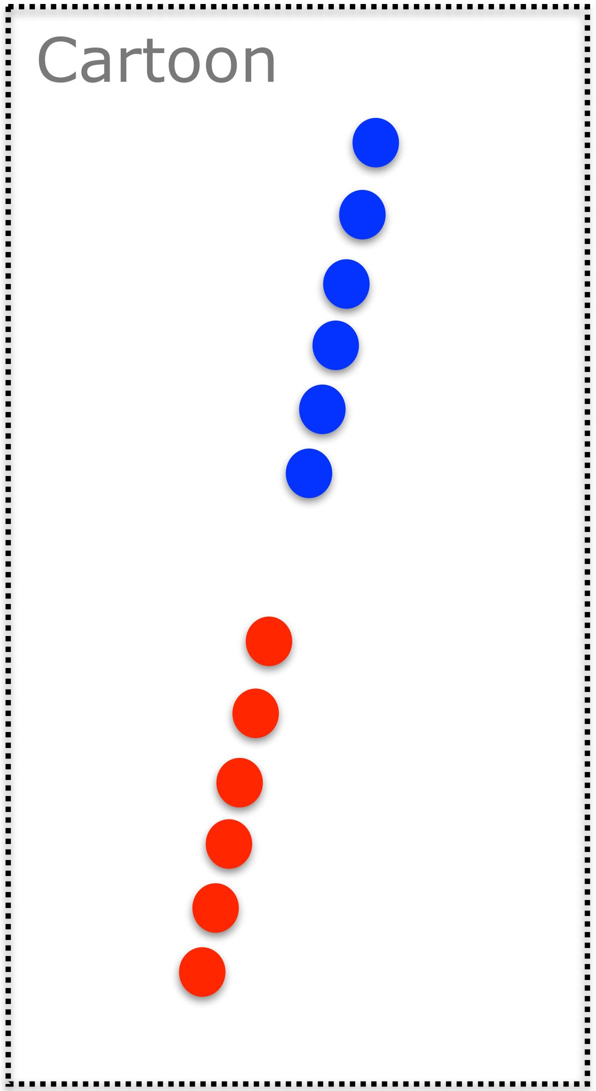
Example: TrackClusterCreation

- Build track-like proto-clusters by joining together “unambiguous” lines of contiguous hits.
- Subsequent track reconstruction depends on identifying some building blocks in this first step.





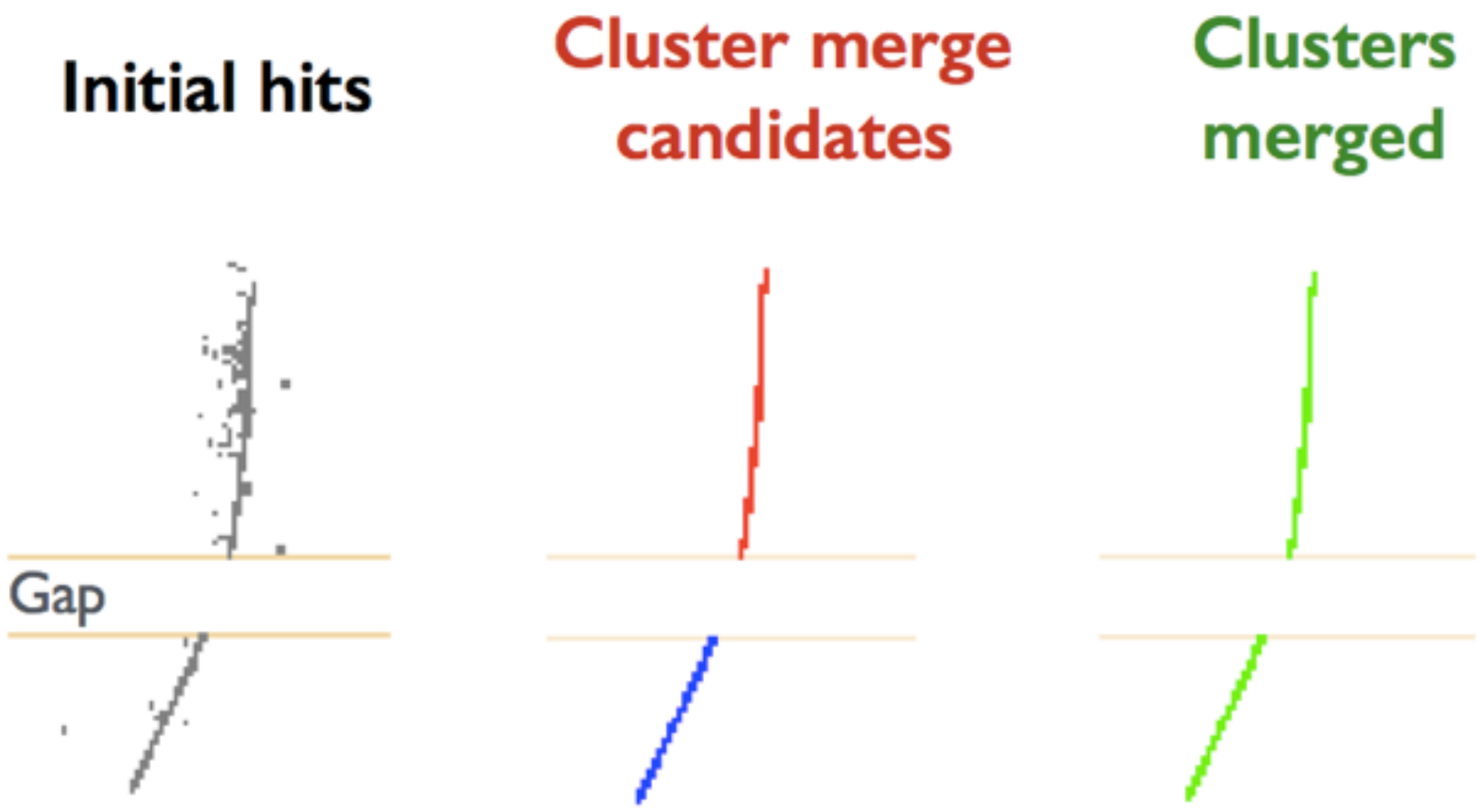
Example: LongitudinalExtension



- Join together track-like proto-clusters separated by small gaps (or small showers).
- Use "ClusterExtension" **design pattern**:
  - ➢ Select "clean" clusters (typically require a minimum cluster length here).
  - ➢ Compare every clean cluster with every other clean cluster.
  - ➢ Identify end-to-end associations between each pair of clusters.
    - In LongitudinalExtension algorithm, apply selection cuts on relative angle and impact parameters.
    - Apply "Sliding Linear Fits" to clusters, to calculate this information.
  - ➢ Select "strong" end-to-end associations and merge these pairs of clusters.

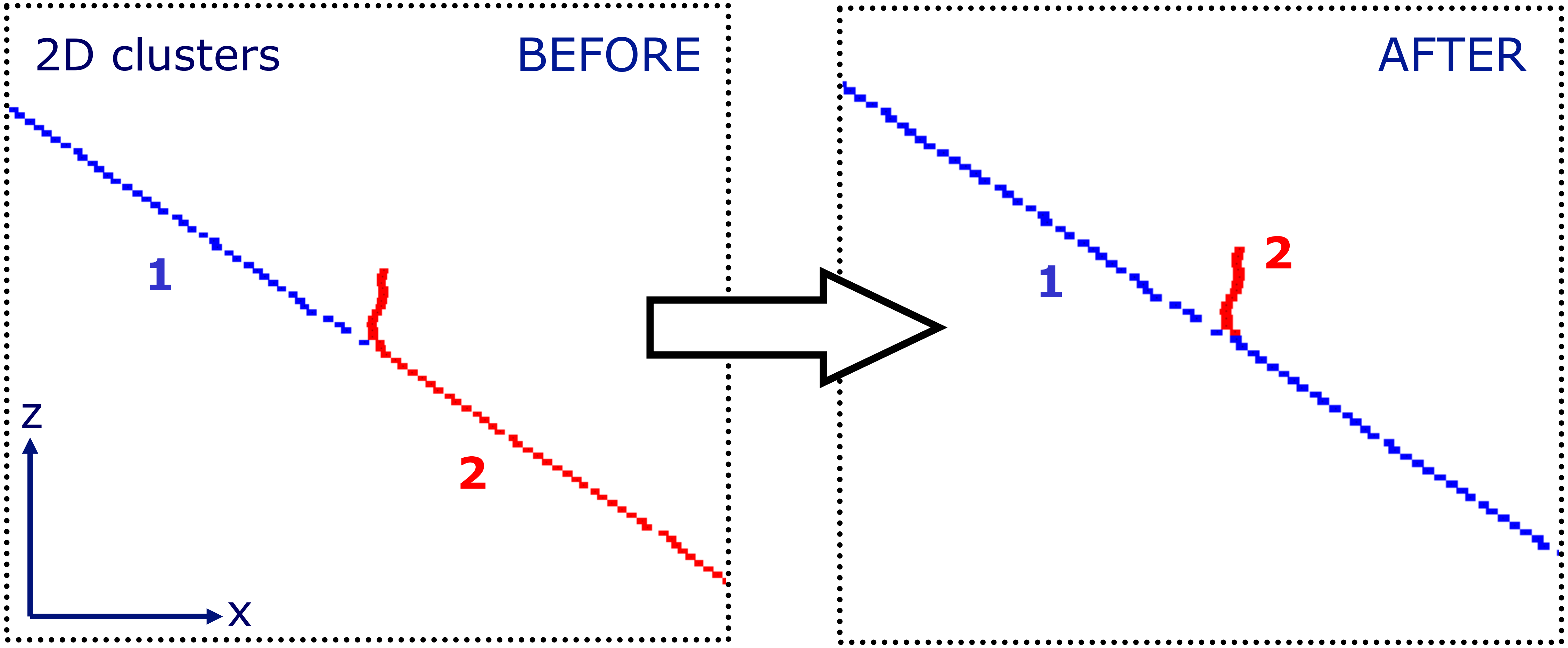
Example: CrossGapsExtension

- A couple of algorithms have been developed to join track segments across detector gaps.
- One of the algorithms, CrossGapsExtension, is based on the same 'ClusterExtension' design pattern.
  - ➢ Select only those clusters that begin or end near to a gap.
  - ➢ Merge together clusters that point at each other across a gap.



Example: BranchSplitting

- Common pathology identified in early pattern recognition: 2D track follows a delta ray rather than the main track.
- Address this using a family of algorithms that "split and splice" track segments into a single track, as illustrated below:



2D Shower Reconstuction

Neutrino Pass

(2D showers)

ClusterCharacterisation

ShowerGrowing

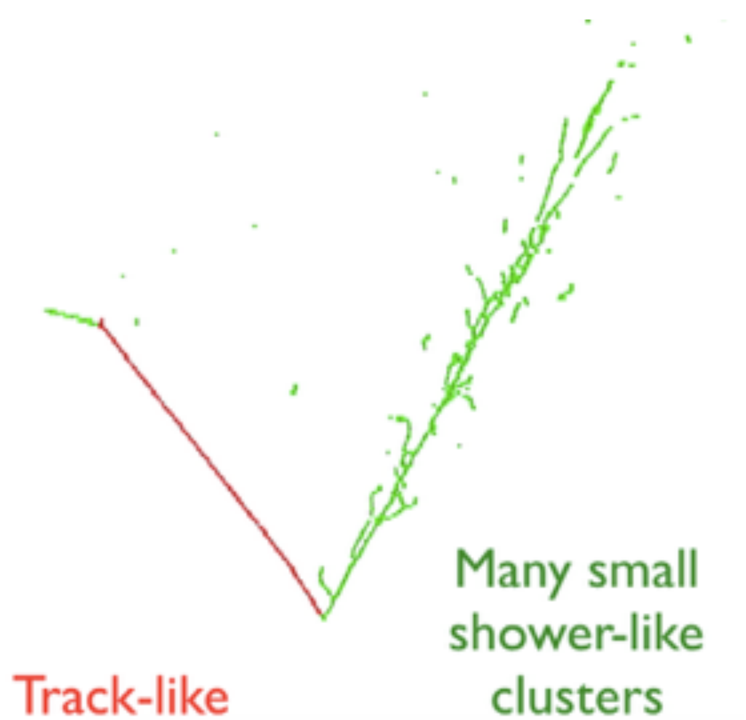
1. Characterise individual clusters as track-like or shower-like, based on topological information.
   - ➢ Use cuts placed on cluster length, transverse width, sparsity of hits.

2(a) Select long shower-like clusters to act as 2D shower seeds.
   - ➢ These represent "shower spines".

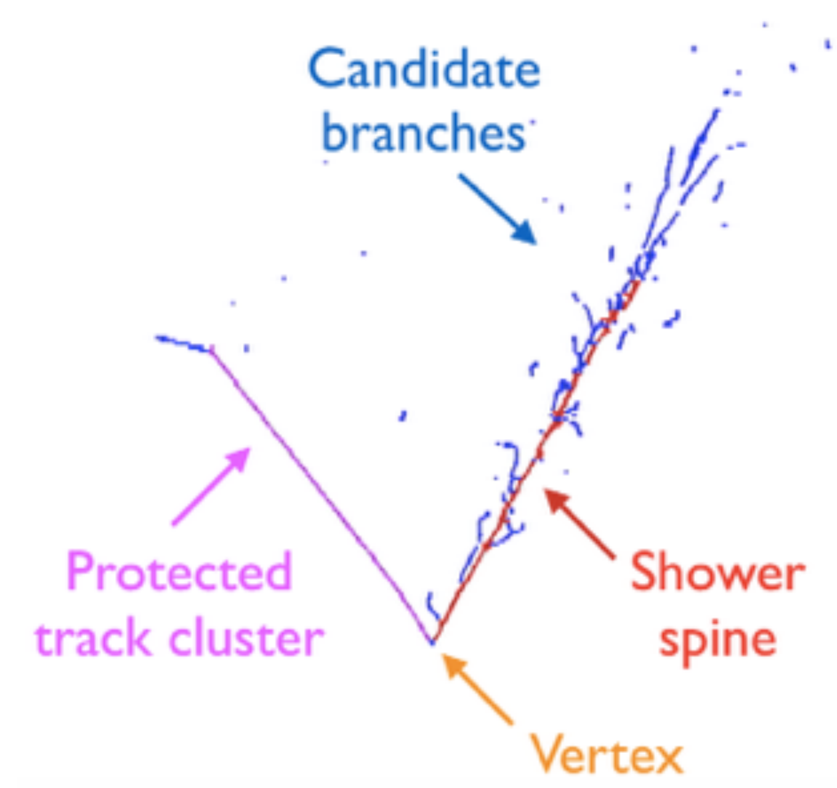
2(b) Addition of shower-like cluster "branches" to shower spines.
   - ➢ Works recursively, finding branches for spines, branches on branches etc., and exploring different choices of top-level spines.

2D Shower Reconstruction

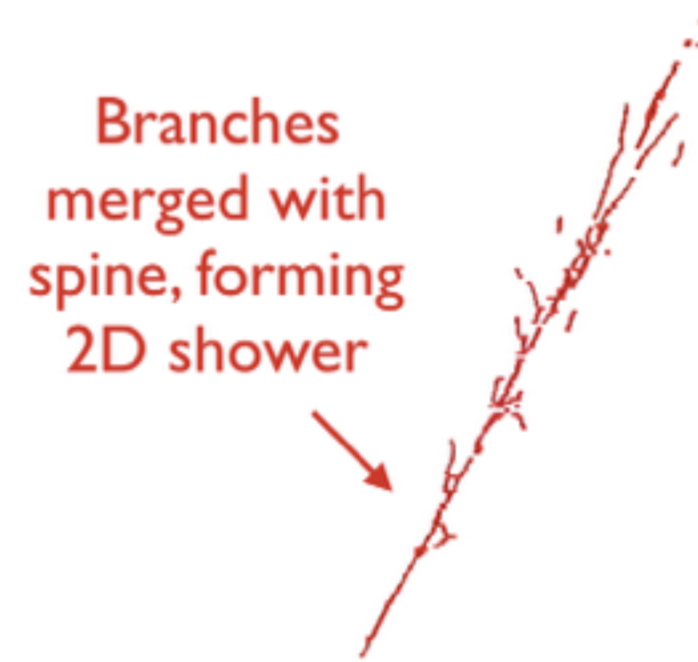
Step 1:
Cluster characterisation



Step 2(a):
Spine identification



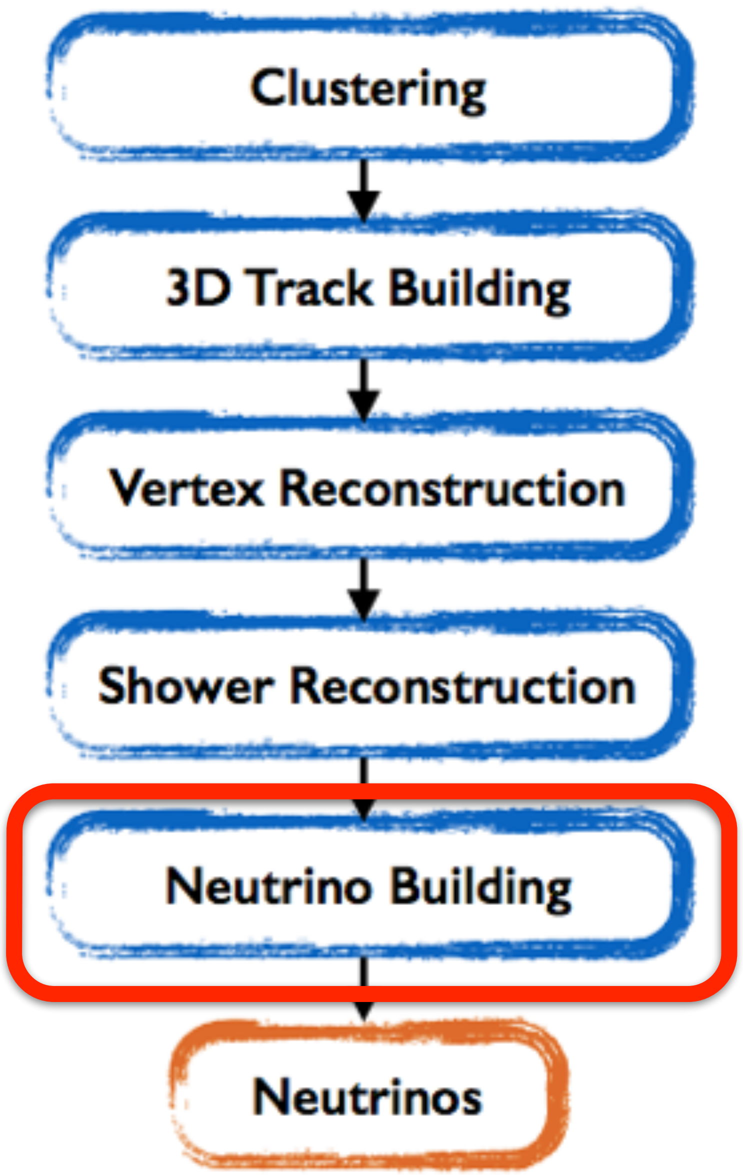
Step 2(b):
Branch-growing



- All clusters are initially classified as “track-like” or “shower-like”.
- Interaction vertex and track-like clusters are “protected” in subsequent shower reconstruction (don’t merge across vertex, don’t merge tracks).
- Select shower spines and run recursive shower-building algorithm.

2D Mop-up Algorithms

Neutrino Pass



- One addition class of 2D algorithm: “mop-up” algorithms.
  - ➢ Identify and merge in extraneous hits, ensuring that reconstructed 3D particles have a complete set of hits
- A suite a 2D mop-up algorithms runs after 3D track and shower building, just prior to “neutrino building”.

← 2D “Mop-up” algorithms

Note: cosmic pass has a similar but simpler set of algorithms designed for delta rays.

➢ Won’t discuss cosmic mop-up here.

2D Mop-up Algorithms

<u>Neutrino Pass</u> (2D mop-up):

BoundedClusterMerging: Addition of hits within a 2D bounding box.

ConeBasedMerging: Addition of downstream hits using a 2D cone centred on the vertex.

ProximityBasedMerging: Addition of nearby hits to nearest 2D cluster.

Finally:

IsolatedHitMerging: Addition of more distant 2D "isolated hits".

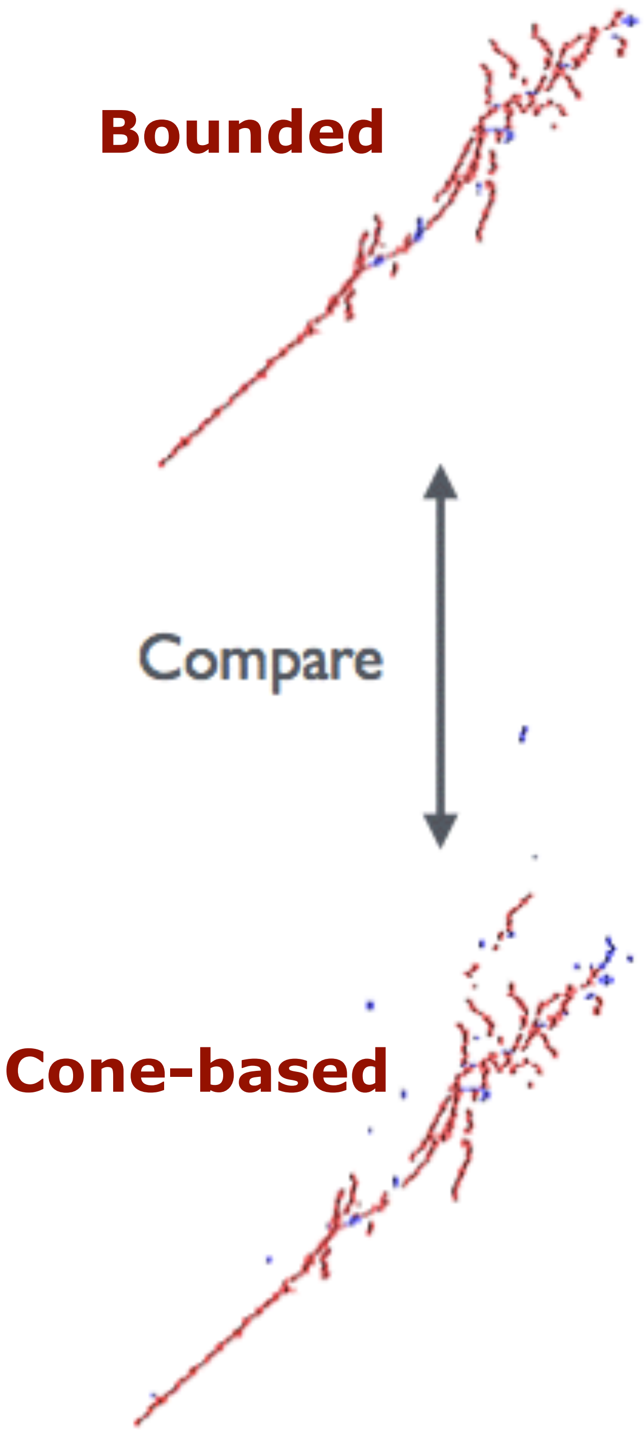
(In the Pandora framework, hits flagged as "isolated" contribute to the energy variables but not topological variables).

Note:

- Prior to 2D mop-up, we also run 3D "particle recovery" algorithms e.g. searches for new particles around the interaction vertex.

2D Mop-up Algorithms

Examples:



- Bounded cluster merging:
  - ➢ Pick up clusters enclosed in shower envelope.
  - ➢ This is a pretty reliable and safe algorithm, hence run this algorithm first.
- Cone-based cluster merging:
  - ➢ Picks up downstream clusters using a cone extended from the vertex.

Note: The mop-up algorithms are intended to collect up the few remaining hits, after all particles have been reconstructed.

- ➢ Indeed, this is mainly what they do!
- ➢ However, sometime these algorithms rescue a large portion of missing hits (which is good) or merge together two particles if one has not been reconstructed (which is bad).

Summary

- 2D pattern recognition is a key part of Pandora pattern recognition. Needs to perform well for overall success!
  - However, we also use 3D information to improve 2D clustering (this is another key part of Pandora pattern recognition).
- There are many different types of 2D reconstruction problem:
  - Cosmic, neutrino, track, shower.
  - Also: non-accelerator, low-energy, high-energy etc…
- Multi-algorithm approach is powerful in developing 2D reconstruction.
  - Build up event gradually using many different algorithms.
  - Have developed common algorithms relevant for more than one types of problem, and dedicated algorithms for specific types.
  - Supported by "design patterns".
- Obtain reasonable performance - but 2D chain is far from finished, and performance is far from perfect.
  - Many possibilities for future development!