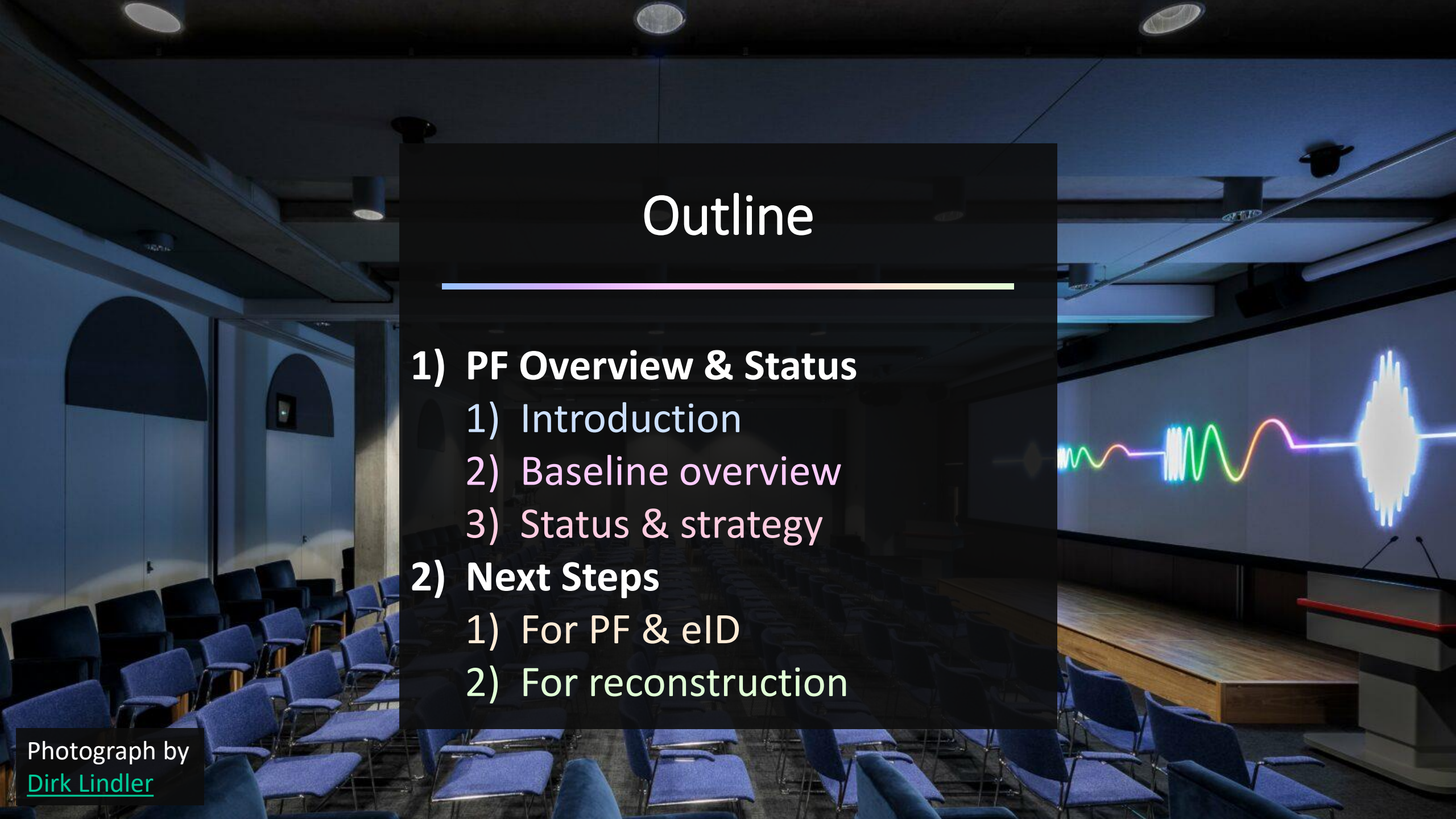
PF Status & Next Reconstruction Priorities

ePIC /EIC Physics Readiness Workshop 2025

Derek Anderson
Jefferson Lab

Photograph by
Dirk Lindler



Outline

1) **PF Overview & Status**
    1) Introduction
    2) Baseline overview
    3) Status & strategy
2) **Next Steps**
    1) For PF & eID
    2) For reconstruction

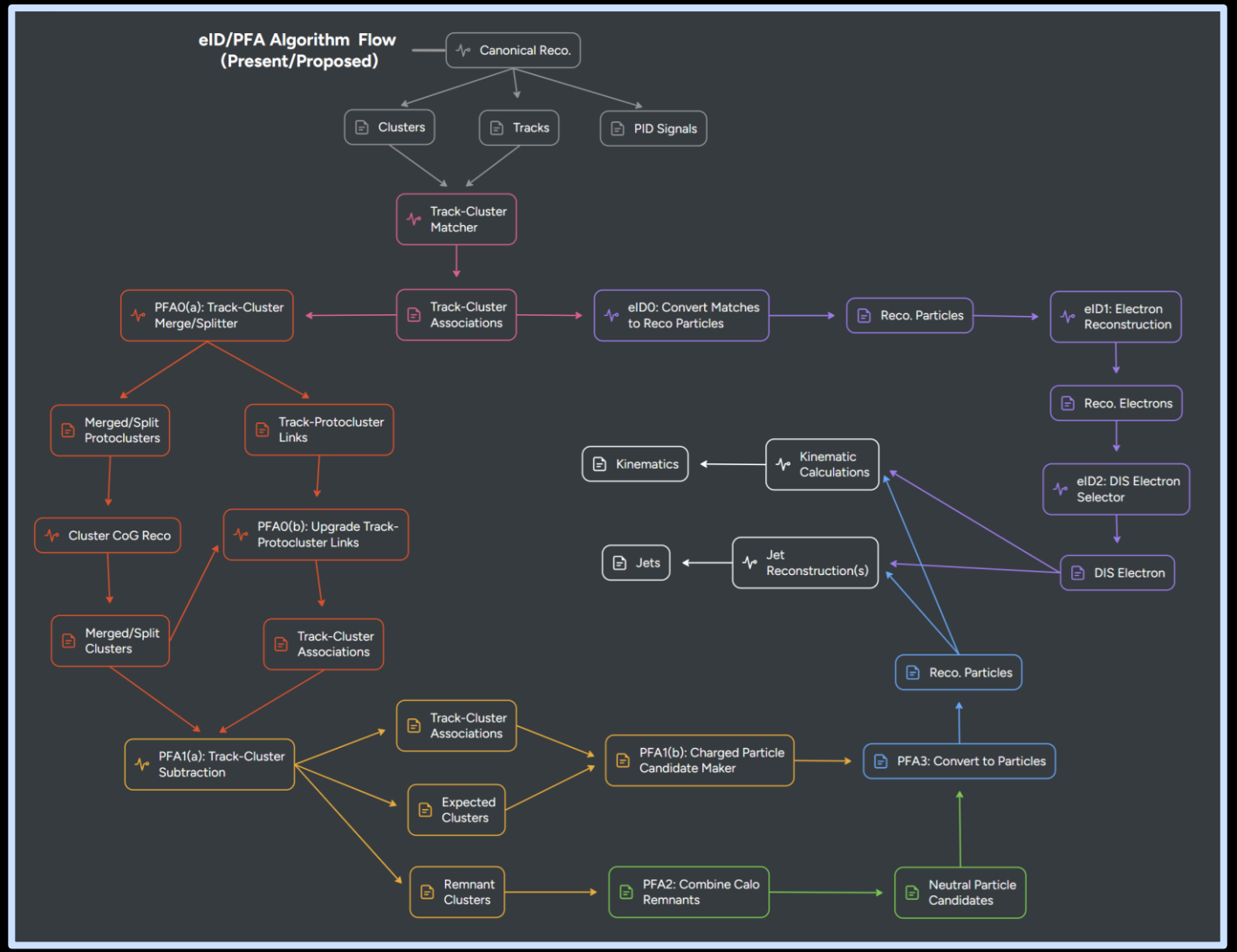
Photograph by Dirk Lindler

Introduction | Particle Flow Task Charge and Goal

- **PF Reconstruction Task:** improve jet reconstruction using particle flow (PF) info
  - But PF also touches many aspects of holistic reco beyond jets
  - eg. Neutral reco benefits greatly from PF techniques

⇒ **Task Goal:** implementation of ***PFAlpha***, a (relatively) simple PF baseline to gauge further developments
  - **Right:** schematic of algorithm flow of PFAlpha
    - Includes implemented, in-progress, and planned stages
  - Aiming to ensure modularity of overall algorithm

- **Note:** PFAX = "PFAlpha stage X"
  - Flow is split up into major stages of the overall algorithm

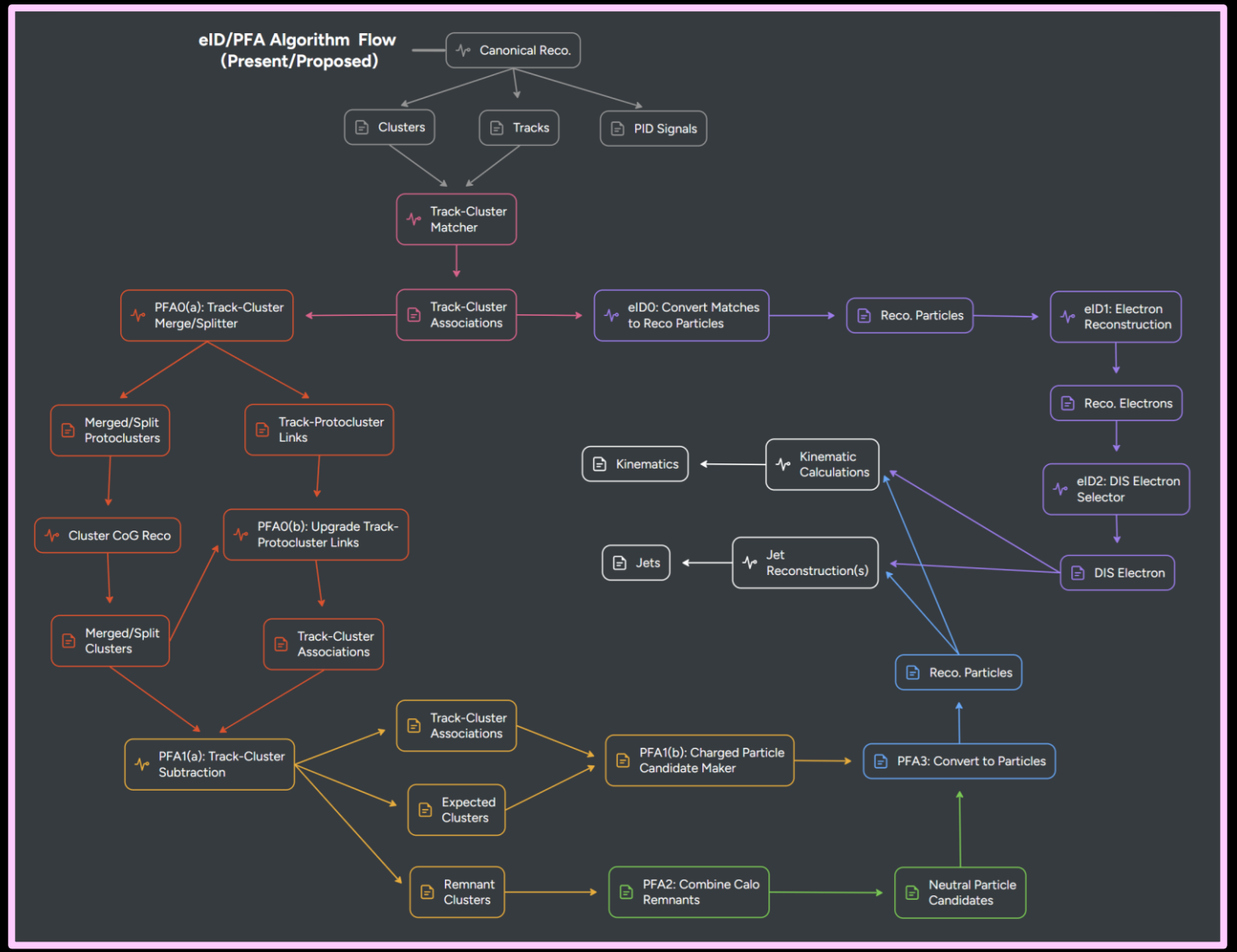


Baseline Overview | Baseline Overview

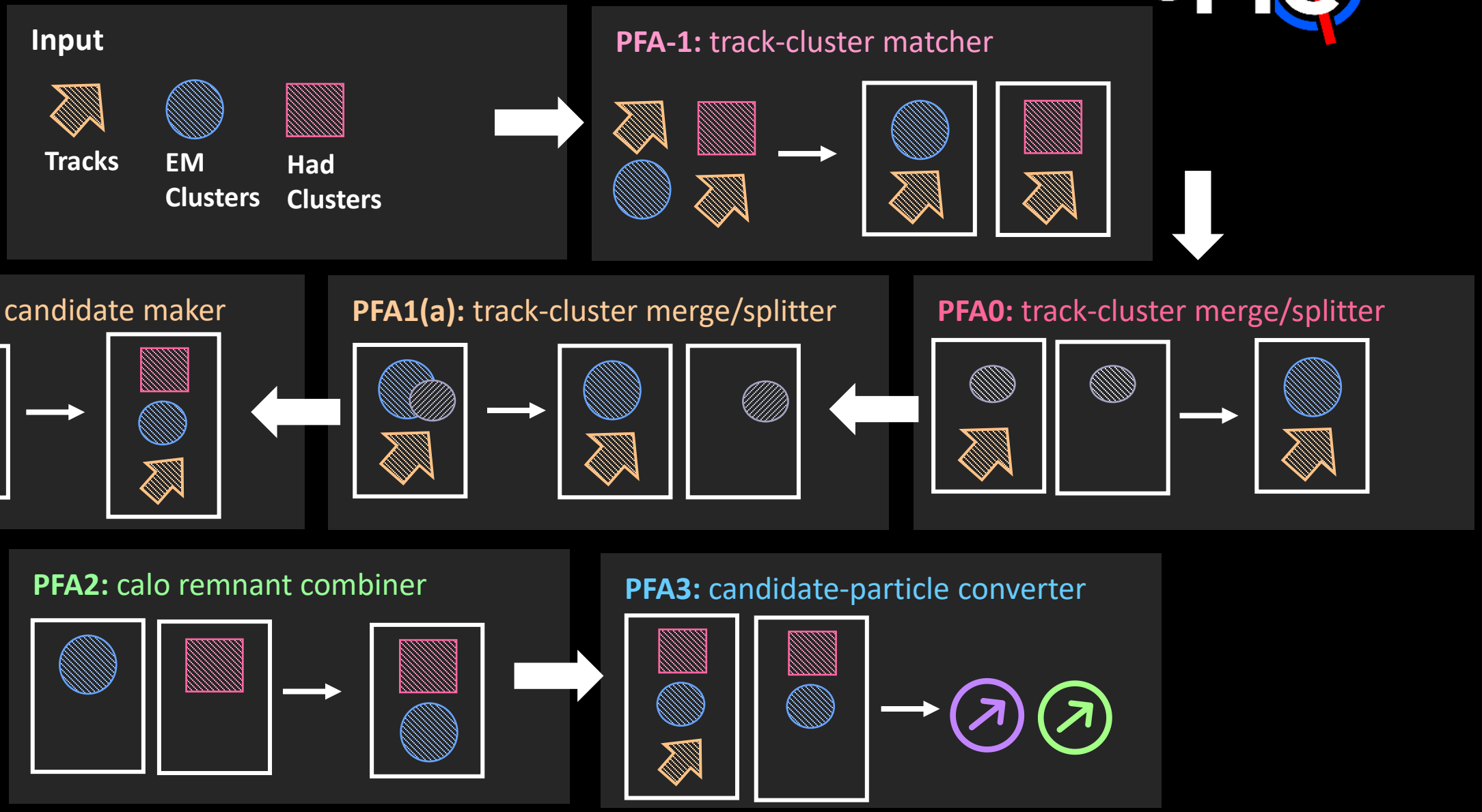
- **In broad strokes:** the overall algorithm is

1) **[PFA-1]** Match tracks to EMCal, HCal clusters
2) **[PFA0]** Merge clusters based on track E/p in a cone of size R_0
   - Split merged clusters between matched tracks
3) **[PFA1a]** Subtract expected track energy from merged clusters
   - Split into tracks + expected energy, and remnant clusters (leftover energy)
4) **[PFA1b]** Convert tracks + expected energy to particle candidates
5) **[PFA2]** Combine remnant EMCal, HCal clusters in a cone of size R_1, convert to particle candidate
6) **[PFA3]** Convert candidates to reconstructed particles

- **Note:** finer details of each stage in backup



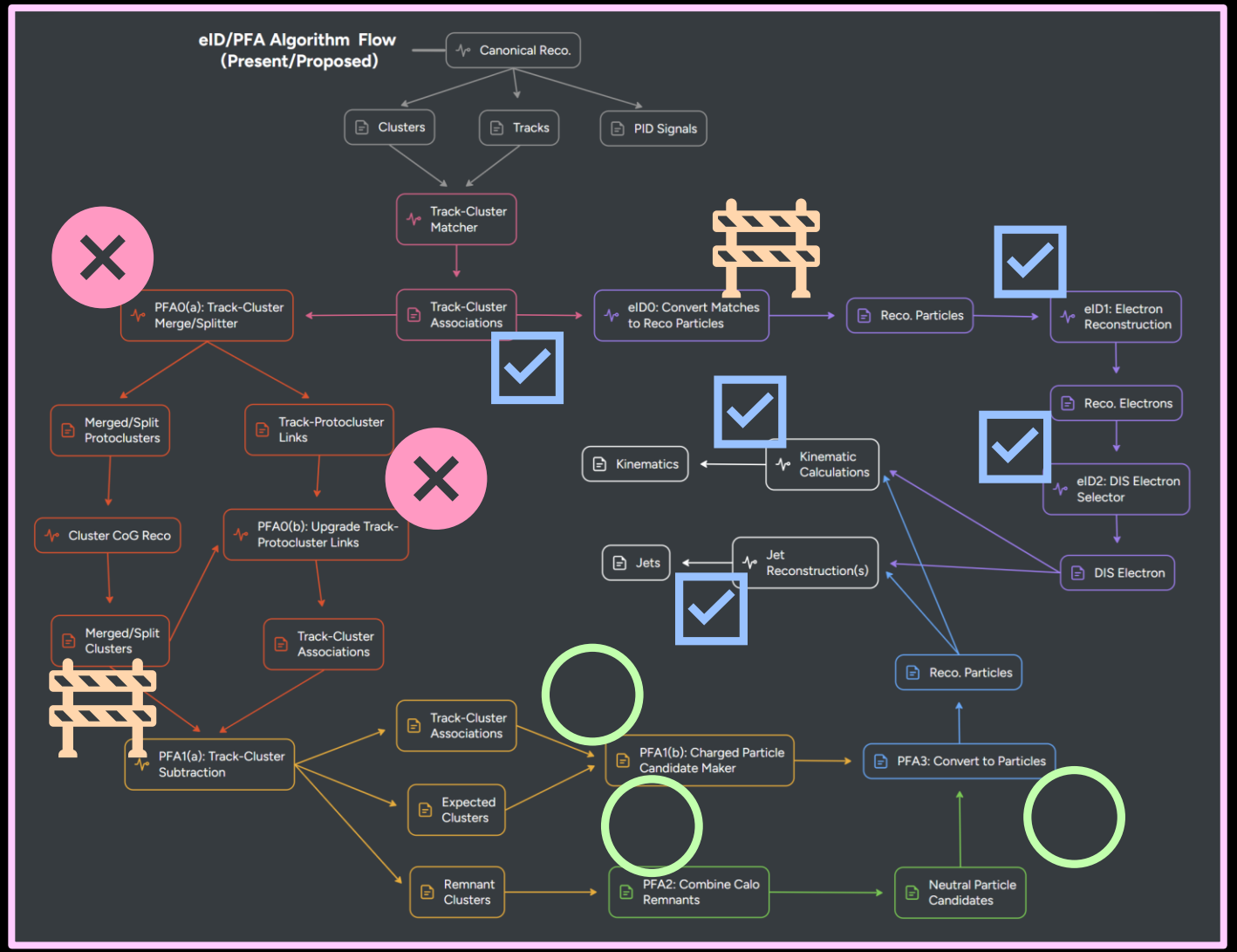
Baseline Overview | Schematic Representation



Status & Strategy | Current PF Status (1/2)

- **Current status:** in development...
  - Icons indicate status of each stage
- **Focus at start of year:** PFA0 (merge/splitter) and PFA-1 (track-cluster matcher)
  - PFA-1 implementation complete at start of July in EICrecon#1906
  - Merge/splitter algorithm already implemented last year,
  - ***But* needs update to pass track associations downstream**

○ = To-do
🚧 = In progress
☑ = Done/already in EICrecon
⊗ = Blocked

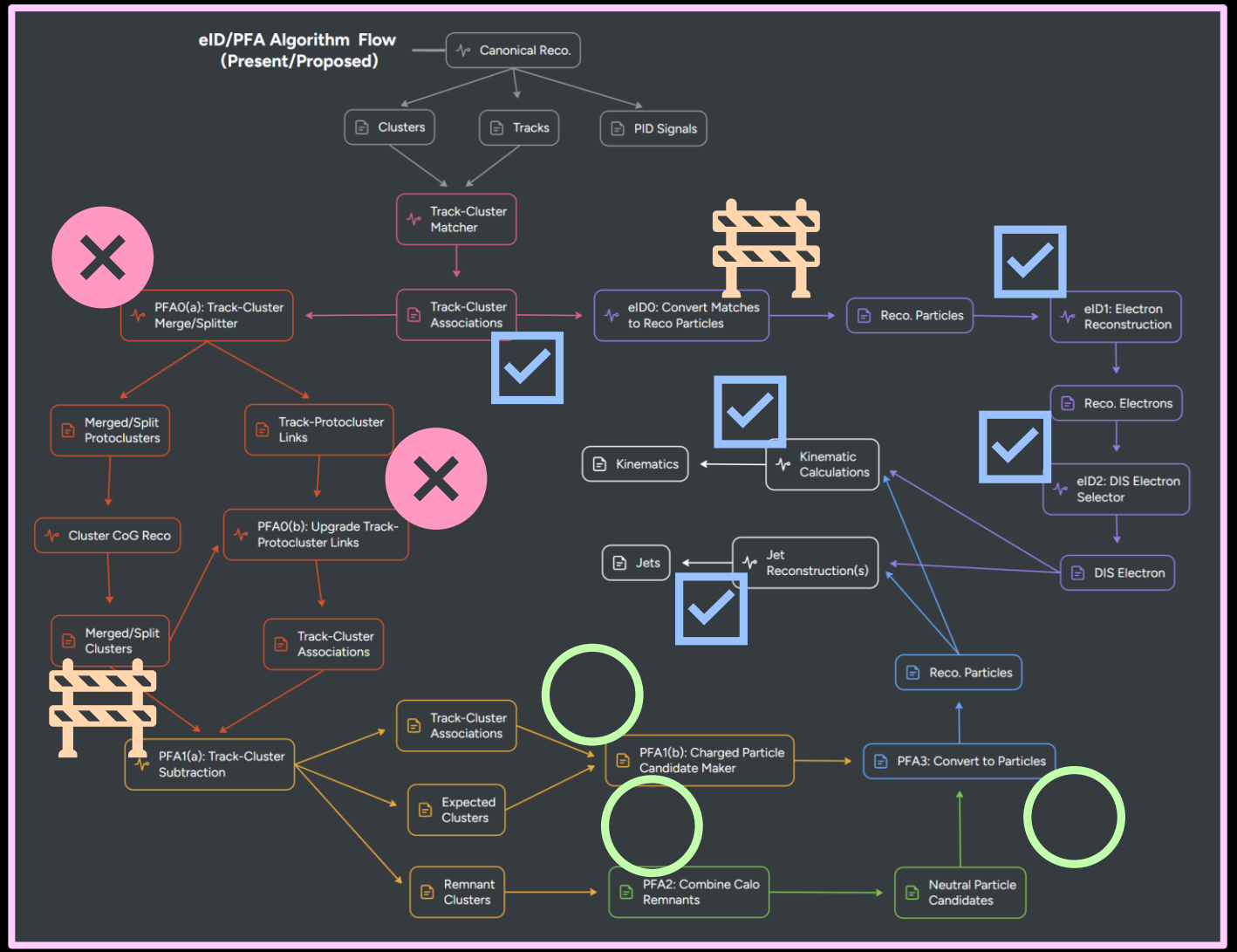


Status & Strategy | Current PF Status (2/2)

- Strategy for PFA0 required data-model change, track-protocluster links
  - which exposed bug in our JANA2 extensions
  - Now blocked until we upgrade to JANA2 2.4.3 (which has breaking changes)

- Since then, worked out a development strategy that
  1) Defers JANA2 2.4.3 upgrade as long as possible
  2) Avoids overly hasty data-model changes

○ = To-do
🚧 = In progress
☑ = Done/already in EICrecon
⊗ = Blocked



Status & Strategy | Proposed Development Strategy

- **Proposal Goal:** *actually* have implementation in by next collaboration meeting
  - ie. Algorithms fully implemented in EICrecon w/ some parameter tuning to-do
    - Benchmarks being reviewed/merged
  - ☞ **Will write technical note after implementation complete**

- **Only doable if we have additional people doing development!**
  - More people means we can parallelize development!
  - **Development threads:**
    - **PFA-1**
    - **PFA0**
    - **PFA1**
    - **PFA2**
    - **PFA3**

- **Proposal:**
  1) Flag EMCal vs. HCal clusters with Cell ID for now (see backup)
     - **Target:** 25.10.0
  2) Develop threads proceed in parallel, aiming to complete at roughly same time
     - Each developer ***also* creates, submits benchmark** for thread (see backup)
     - **Target:** 25.12.0
  3) Final PR to tie threads together into PFAlpha
     - **Target:** 26.01.0

☞ **Note:** targets listed are target campaigns, erring on cautious side

Status & Strategy | Task List

Tasks	Issue/PR/Note	Est. labor time*	Assignee
PFA-1/eID0: deprecate MatchClusters, replace w/ pure reco equivalent	EICrecon#1956	2 weeks	Tristan
PFA0(a): complete merge/splitter update (requires JANA2 2.4.3)	EICrecon#1699	1 week	**BLOCKED**
PFA0(b): implement track-protocluster link promotion algorithm	EICrecon#1886	2 weeks	**BLOCKED**
PFA1(a): revive and finish track-cluster subtractor	EICrecon#1627	1 week	Derek
PFA1(b): track-cluster converter (synergy w/ PFA-1)	To-do	1 week	**OPEN**
EDM: flagging ecal vs. hcal clusters	EICrecon#2078	1 week	Derek
PFA2: implement calo remnant combiner	To-do	2 weeks	**OPEN**
PFA3: implement particle regressor/convertor	To-do	2 weeks	**OPEN**

* Assuming 50% FTE, including code review time

☞ **Note:** associated benchmark tasks in backup

Next Steps | For PF and eID (1/2)

- **Immediate task:** complete baseline implementation
  - The more we can parallelize, the faster it'll go!
  - Lack of some PF machinery quickly becoming pain point

- Several directions after PF can move after baseline
  - **Near(ish) term book-keeping tasks**
    - Ensure track-cluster matches used where relevant
    - Upgrade Track-Cluster Matches from *associations* to *links* (after JANA2 2.4.3)
  - **Future steps towards *PFBeta*, *PFGamma*, etc.**
    - Integrating PID info
    - Topoclustering *across* calorimeters
    - AI/ML can be integrated in several stages
    - Specialized PF algos for high-granularity calos

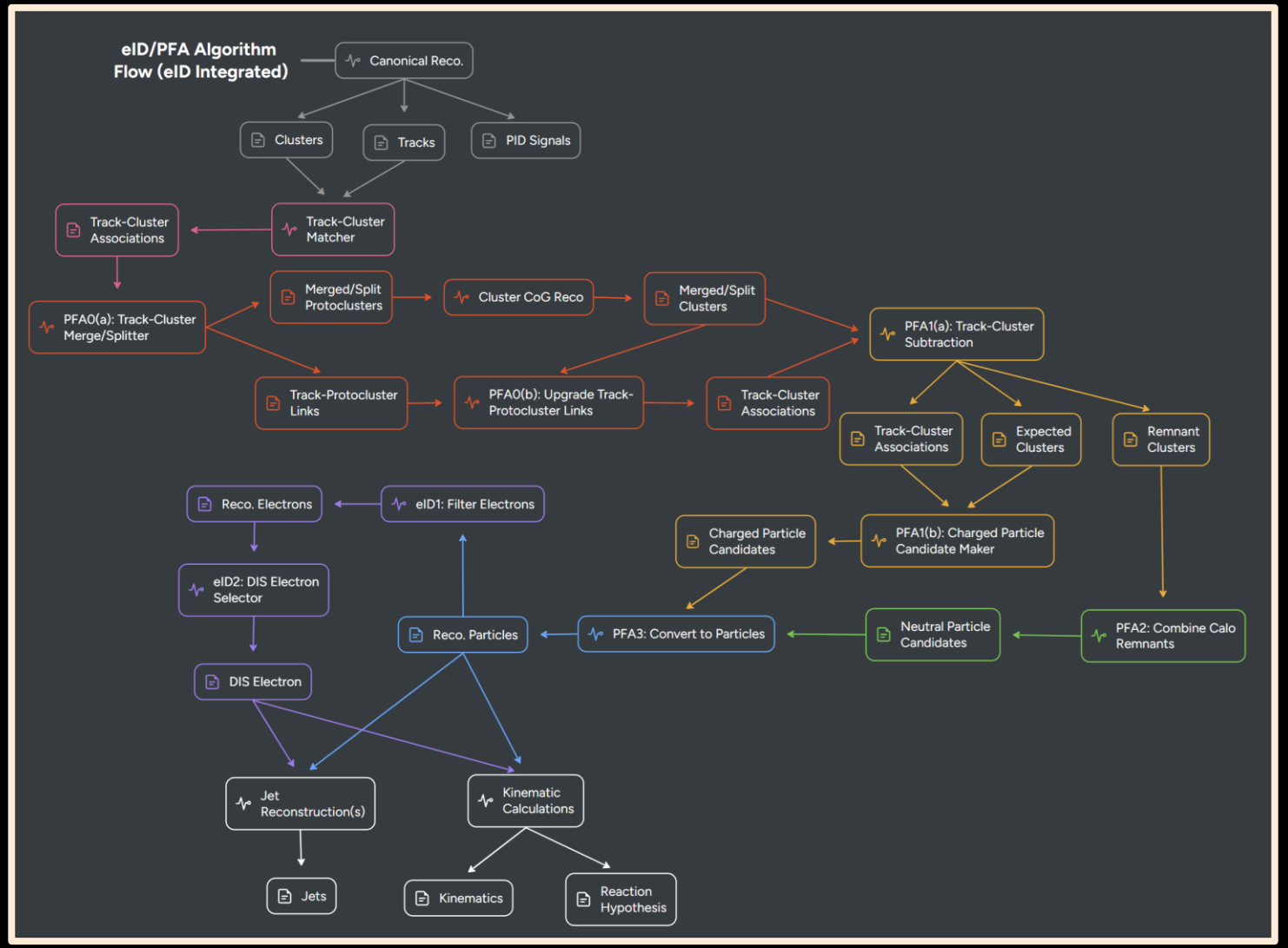
- **Critical next step:** integrating PF baseline and electron ID (eID)
  - ☞ Once PFAlpha implemented and vetted, can easily wire together PF and eID (next slide)

- How does this relate to the standalone tools/analyses?
  - Personally
    - ⇒ **Anything recommended to collaboration as a whole should run by default in a common framework (ie. EICrecon)**
    - ∴ PFAlpha + eID should produce reconstructed particles + DIS e- that will work for most user analyses
  - **But flexibility to prototype/sandbox is important!**
    - ☞ To discuss: is there a way to deploy eID algorithms in both contexts?

Next Steps | For PF and eID (2/2)

- **Initial steps towards integrating PFAlpha and eID:**
  1) Replace ElectronReconstruction with algorithm to filter e- from PFA3 output
     - ⇒ PFA3 produces Reconstructed Particles with rough PID
     - ∴ Will duplicate some functionality of existing e-reco algorithm
  2) Filtered e- will be passed to ScatteredElectronsEMinusPz to obtain same ordered list of DIS e- candidates

- **Future directions**
  1) Expand PFA3/e- filter to include more sophisticated criteria (eg. PID info, isolation, etc.)
  2) Expand DIS e- selector to accommodate standalone developments & ***multiple selection algorithms***



Next Steps | For Reconstruction

- **This slide:** some (mostly) off-the-cuff thoughts about potential tasks that could be future priorities (or part of future priorities)
  - Includes both potential physics tasks, and tasks tangentially related to physics
  - Fairly biased (apologies!)

- **Some thoughts:**
  - PID development
    - And integration into holistic reco
  - Generalized resonance reconstruction (Lambdas, D0, etc.)
  - Background impact on holistic reconstruction
  - Timeframe integration

- **More thoughts**
  - PF/eID evolution
    - Integrating PF and eID (previous slide)
    - Enhancing eID (previous slide)
    - Generalized topoclustering
  - Clustering
    - Consolidation/update of existing algorithms
  - Expansion of kinematic/inclusive algorithms
    - Development of a ReactionHypothesis type + algorithm (cf. Aug. 4th Reco WG discussion)
    - Tighter integration of central & FF/FB regions

Open Discussion

- **PF**
  - PID integration
  - Generalized topoclustering *across* calorimeters
  - AI/ML integration
  - Specialized PF algos for high-granularity calos
- **eID**
  - Deploying eID algorithms in both EICrecon + standalone contexts
  - Integrating PFAlpha + eID
  - PFA3/e- filter expansion
  - DIS e- accommodation of multiple selection algorithms
- Generalized resonance reconstruction (Lambdas, D0, etc.)
- Background impact on holistic reconstruction
- Timeframe integration
- Clustering
  - Consolidation/update of existing algorithms
- Kinematic/inclusive algorithms expansion
  - Development of a ReactionHypothesis type + algorithm
  - Tighter integration of central & FF/FB regions

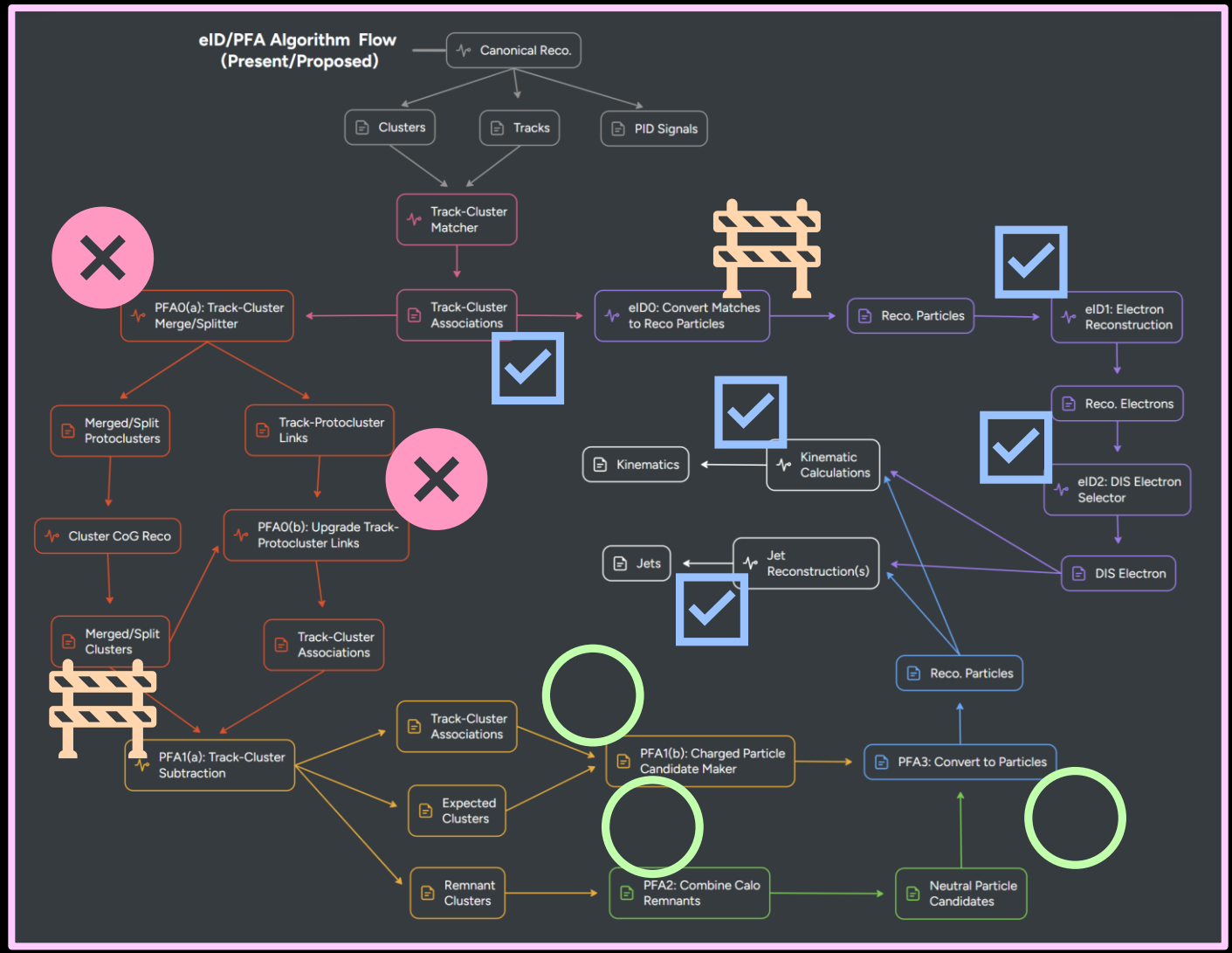
Questions, thoughts, or suggestions about reconstruction? Ask the WG with the form here!

Photograph by Dirk Lindler

Backup | Timeline Since Frascati

Developments since Frascati CM:

1) [02.06.2025] PFA0 PR opened and in progress
2) [02.23.205] PFA1(a) PR ready to open after PFA0 merged
3) **[02.27.2025]** Held 1st PF SubWG meeting
4) **[03.11.2025]** Draft PR for candidate types open
5) **[03.28.2025]** Initial track-cluster matcher merged
6) **[05.06.2025]** Track-Protocluster link merged
7) **[07.06.2025]** Multi-calo track-cluster matcher merged
8) **[08.01.2025]** JANA2 2.4.3 released
9) **[08.05.2025]** Held 1st PF Tech Discussion
10) **[08.28.2025]** PFA1(a) open for review
11) **[09.02.2025]** New development plan proposed
12) **[09.15.2026]** EMCal-HCal flag PR opened and in progress



Backup | PFA-1/eID0 | Track-Cluster Match Converter

- **Track-Cluster Match Converter:** MatchClusters (current source of ReconstructedParticles) is one of biggest truth info leaks in reconstruction
  - But with track-cluster matches, we can now patch it (see EICrecon#1956)
  - Intended to supersede MatchClusters while PFAlpha develops

- **The algorithm:** is identical to MatchClusters, but with track-cluster matches rather than truth info
  1) Build map of tracks onto matched clusters, and set of clusters
  2) For each track:
     a) Remove matched cluster from cluster set
     b) Copy associated charged particle of track into output
  3) For each remaining cluster in set:
     - Create a reconstructed particle with mass and PDG of 0

Inputs:
- Track-cluster matches
- Reconstructed charged particles
- Clusters
  - Planned on only using EMCal clusters (like MatchClusters)
  - But *could* add in HCal's

Outputs:
- Reconstructed particles

Parameters:
- None (if using only EMCal clusters)

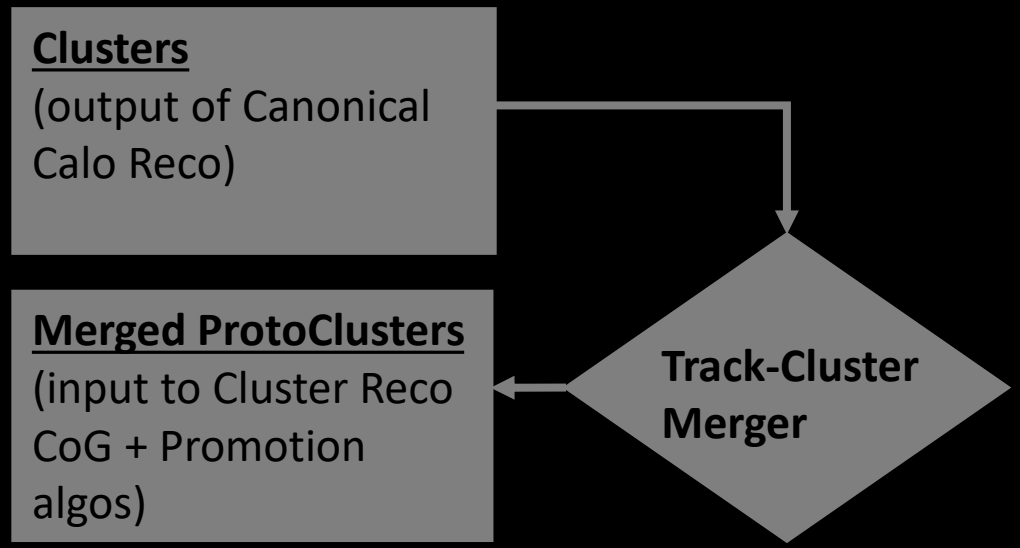
Backup | PFA0 | Track-Cluster Merge/Splitter (1/2)

- **Track-Cluster Merging:** implemented to address in pTDR need (cluster merging)
  - Algorithm outline based on ATLAS's split recovery procedure
    - c.f. Eur. Phys. J. C (2017) 77:466
  - Implemented in EICrecon#1406

- **The algorithm**
  1) Match track projection to cluster
  2) If matched, calculate significance b/n E_{clust} energy & expected E_{dep}:

$$S(E_{clust}) = \frac{E_{clust} - (p_{proj} \times \langle E/p \rangle)}{\sigma(E_{dep})}$$

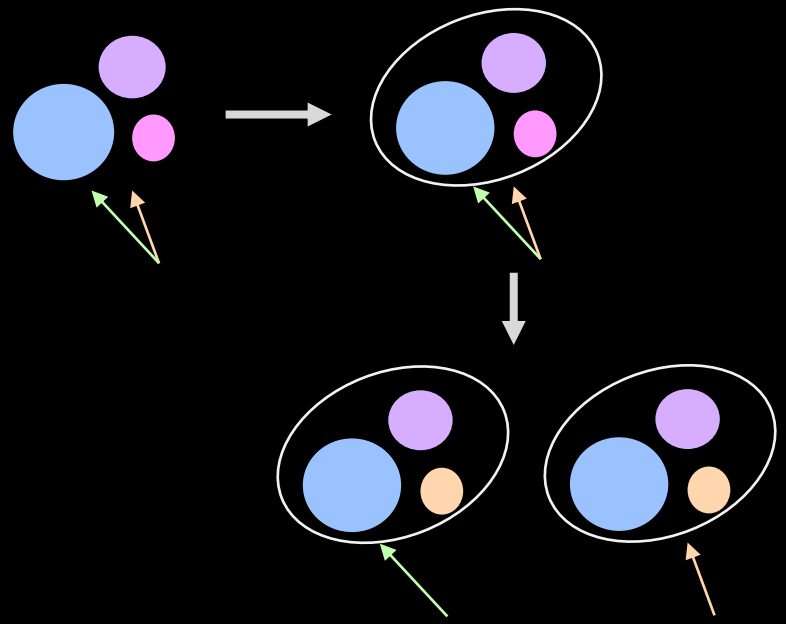
  3) If $S < S_{cut}$, add clusters inside Δr_{add}
  4) If multiple tracks pointing to merged cluster:
     - Split into one cluster for each track & reweight transverse shape by p_{trk}



Parameters:
- $\langle E/p \rangle$: Average E/p
- $\sigma(E_{dep})$: Spread of dep. energy
- S_{cut}: Threshold to run split-recovery
- Δr_{add}: Window to add clusters
- σ_{trk}: scale for transverse shape reweighting

Backup | PFA0 | Track-Cluster Merge/Splitter (2/2)

- **Track-Cluster Merging:** implemented to address in pTDR need (cluster merging)
  - Algorithm outline based on ATLAS's split recovery procedure
    - c.f. Eur. Phys. J. C (2017) 77:466
  - Implemented in EICrecon#1406



- **The algorithm**
  1) Match track projection to cluster
  2) If matched, calculate significance b/n E_{clust} energy & expected E_{dep}:

$$S(E_{clust}) = \frac{E_{clust} - (p_{proj} \times \langle E/p \rangle)}{\sigma(E_{dep})}$$

  3) If $S < S_{cut}$, add clusters inside Δr_{add}
  4) If multiple tracks pointing to merged cluster:
     - Split into one cluster for each track & reweight transverse shape by p_{trk}

Parameters:
- $\langle E/p \rangle$: Average E/p
- $\sigma(E_{dep})$: Spread of dep. energy
- S_{cut}: Threshold to run split-recovery
- Δr_{add}: Window to add clusters
- σ_{trk}: scale for transverse shape reweighting

Backup | PFA1(a) | Track-Cluster Subtractor (1/3)

- **Track-Cluster Subtractor:** subtracts momentum of matched track(s) from cluster
  - In progress at EICrecon#1627

- **The algorithm:**
  1) Build map of clusters onto *all* matched tracks
  2) For each cluster:
     a) Sum energy of matched tracks:

$$E_{trk} = \sum p_{trk}(S_{use}) \oplus m_{trk}$$

     b) Subtract sum: $E_{sub} = E_{clust} - f_{sub}E_{trk}$
     c) If NOT consistent w/ 0,
        - Create remnant cluster w/ E_{sub}
        - Set expected cluster energy to $E_{clust} - E_{sub}$
     d) Create an association for each track matched to expected cluster

Inputs:
- Track-cluster matches
- Clusters
- Track projections

Outputs:
- Remnant clusters ($E_{sub} = E_{clust} - E_{trk}$)
- Expected clusters ($E_{clust} - E_{sub}$)
- Track-expected cluster association

Parameters:
- f_{sub}: fraction of track energy to subtract
- $m_{default}$: default mass to use for track energy
- S_{use}: surface to evaluate track momentum at
- $k_{do\ n\sigma?}$: turn on/off checking against resolutions
- $n\sigma_{cut}$: max no. of sigmas to be consistent w/ 0
- σ_{trk}: tracking resolution to use in n-sigma cut
- σ_{cal}: calo resolution not use in n-sigma cut

Backup | PFA1(a) | Track-Cluster Subtractor (2/3)

- **Track-Cluster Subtractor:** subtracts momentum of matched track(s) from cluster
  - In progress at EICrecon#1627

- **The algorithm:**
  1) Build map of clusters onto *all* matched tracks
  2) For each cluster:
     a) Sum energy of matched tracks:

$$E_{trk} = \sum p_{trk}(S_{use}) \oplus m_{trk}$$

     b) Subtract sum: $E_{sub} = E_{clust} - f_{sub} E_{trk}$
     c) If NOT consistent w/ 0,
        - Create remnant cluster w/ E_{sub}
        - Set expected cluster energy to $E_{clust} - E_{sub}$
     d) Create an association for each track matched to expected cluster

Sub-routine: is E_{sub} consistent w/ zero?

1) **If $E_{sub} < 0$, YES**
2) **Else if $k_{do\ n\sigma?}$**
   a) Calculate $n\sigma$

$$n\sigma = \frac{E_{sub}}{\sigma_{trk} \oplus \sigma_{cal}}$$

   b) **If $n\sigma < n\sigma_{cut}$, YES**
3) **Else**
   a) **If $E_{sub} < \epsilon$, YES**

Note: epsilon here is std::numeric_limits<double>::epsilon()

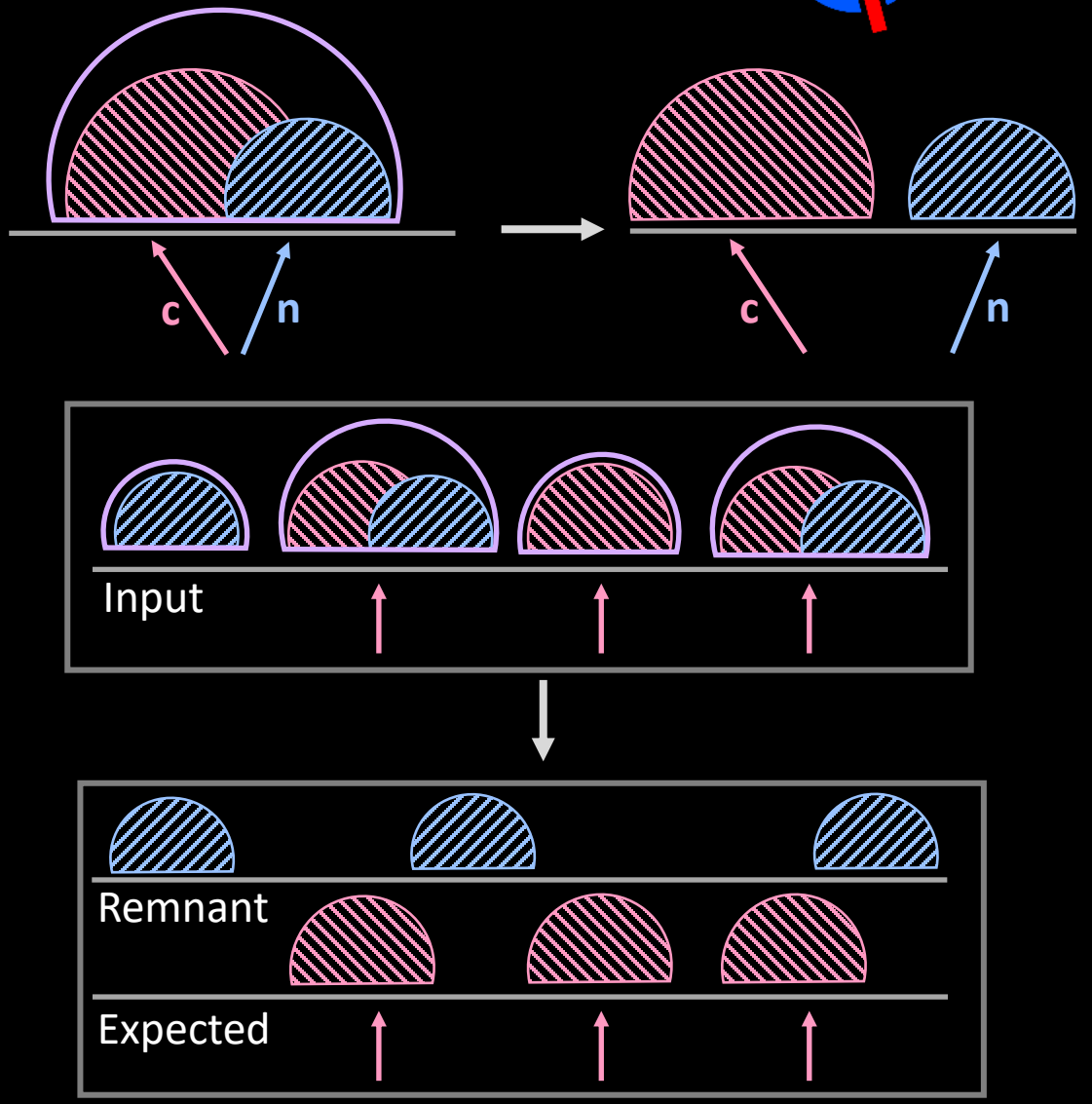
Backup | PFA1(a) | Track-Cluster Subtractor (3/3)

- **Track-Cluster Subtractor:** subtracts momentum of matched track(s) from cluster
  - In progress at EICrecon#1627

- **The algorithm:**
  1) Build map of clusters onto *all* matched tracks
  2) For each cluster:
     a) Sum energy of matched tracks:

$$E_{trk} = \sum p_{trk}(S_{use}) \oplus m_{trk}$$

     b) Subtract sum: $E_{sub} = E_{clust} - f_{sub}E_{trk}$
     c) If NOT consistent w/ 0,
        - Create remnant cluster w/ E_{sub}
        - Set expected cluster energy to $E_{clust} - E_{sub}$
     d) Create an association for each track matched to expected cluster



Backup | PFA1(b) | Charged Particle Candidate Maker

- **Charged Candidate Maker:** forms track-cluster matches into a charged particle candidate
  - To-do

- **The algorithm:**
  1) Build map of tracks onto *all matched clusters*
  2) For each track**:**
     a) For each matched cluster:
        i. Identify if in an ECal or an HCal by checking system ID
        ii. Select relevant weight
        iii. Add to relevant members
     b) Add to relevant member

Inputs:
- Track-cluster matches

Outputs:
- Charged particle candidates

Parameters:
- **$\{ID_{ecal}\}$:** IDs of ECals to look for
- **$\{ID_{hcal}\}$:** IDs of HCals to look for
- **$\{w_{em}\}$:** weights of ECals to use (runs parallel to $\{Id_{ecal}\}$)
- **$\{w_{h}\}$:** weights of HCals to use (runs parallel to $\{Id_{hcal}\}$)

Backup | PFA2 | Calo Remnant Combiner

- **Calo Remnant Combiner:** combines remnant clusters from subtractor into neutral particle candidates
  - Still to-do!

- **The algorithm:**
  1) Combine nearby ECal, HCal clusters
     a) Identify seed ECal cluster
     b) Merge all ECal, HCal clusters in Δr^{em}_{add}, Δr^{h}_{add} of seed and create neutral candidate
     c) Repeat until no ECal clusters are left
  2) Combine remaining HCal clusters
     a) Identify seed HCal cluster
     b) Add all HCal clusters in Δr^{h}_{add} of seed and create neutral candidate
     c) Repeat until no HCal clusters are left

Inputs:
- Remnant ECal clusters
- Remnant HCal clusters

Outputs:
- Neutral particle candidates

Note: maybe make inputs vectors of collections?

Parameters:
- $\mathbf{\Delta r^{em}_{add}}$: window to add ECal clusters
- $\mathbf{\Delta r^{h}_{add}}$: window to add HCal clusters
- $\{w_{em}\}$: weights (or weight) of ECals to use
- $\{w_{h}\}$: weights (or weight) of HCals to use

Backup | PFA3 | Candidate-to-Particle Converter

- **Particle Converter:** takes candidate particles and turns them into reconstructed particles
  - Still to-do!

- **The algorithm:**
  1) Assign preliminary PID based on what info is available (e.g. no hcal clusters → electron, photon, or pi0)
  2) Calculate track energy
  $$E_{trk} = p_{trk} \oplus m_{pid}$$
  3) Calculate calorimeter energy
  $$E_{cal} = N_{cal}\left(\sum w_{em}E_{em} + \sum w_{h}E_{h}\right)$$
  4) If charged particle and $k_{use\ \sigma?}$, calculate resolution-weighted average of E_{cal} and E_{trk}
  5) Calculate remaining kinematics and create reconstructed particle

Inputs:
- Candidate charged/neutral particles
- Primary vertices (for neutral candidates)

Outputs:
- Reconstructed particles

Parameters:
- $k_{use\ \sigma?}$: turn on/off using resolution in energy calculation for charged candidates
- N_{cal}: normalization of calo energy
- σ_{trk}: tracking resolution to use in energy calc
- σ_{cal}: calo resolution to use in energy calc

Backup | Benchmark Task List (1/2)

Tasks	Issue/PR/Note	Est. labor time*	Assignee
PFA-1 Benchmark **- input:** Sum eClust, sum pTrk, nClust, nTrk, E/p matched clusters, sum eGenPar, eGenPar, nGenPar **- output:** Sum eRecPar, eRecPar, ePar, nRecPar, nPar, PES/R of reco pars	To-do	1 week	**OPEN**
PFA0 Benchmark **- input:** Sum eClust, eClust, pTrk, nTrk, nClust, E/p matched clusters **- output:** Sum eSMClust, eSMClust, nSMClust, E/p SM clust, dRct SM	Some work done	1 week	**BLOCKED**
PFA1 Benchmark **- input:** Sum eClust, eClust, sum pTrk, pTrk, nTrk, nClust, E/p matched clusters, sum pChrgPar, pChrgPar, nChrgPar **- output (expected):** sum eEXClust, eEXClust, nEXClust, E/p EX clust, dRct EX **- output (remnant):** sum eREClust, eREClust, nREClust **- output:** sum eEXClust + eREClust	To-do	1 week	**OPEN**

* Assuming 50% FTE, including code review time

- **Notes:**
  - PES/R = Particle Energy Scale/Resolution
  - SM = Split/Merge, EX = Expected, RE = Remnant
  - dRct = distance b/n cluster & matched track

Backup | Benchmark Task List (2/2)

Tasks	Issue/PR/Note	Est. labor time*	Assignee
PFA2 Benchmark **- input:** sum eREClust (EM, H), eREClust (EM, H), nREClust (EM, H), sum eNeuPar, eNeuPar, nNeuPar **- output:** sum eRecPar, nRecPar	To-do	1 week	**OPEN**
PFA3: **- input:** Sum eClust, sum pTrk, nClust, nTrk, E/p matched clusters, sum eGenPar, eGenPar, nGenPar **- output:** Sum eRecPar, eRecPar, ePar, nRecPar, nPar, PES/R of reco pars	To-do	1 week	**OPEN**
PHYS Benchmark: JES/R	To-do (just need wiring)	3 days	Dener
PHYS Benchmark[a,b]: Jets - E, mass, FFs (jt, z), Substructure (dRcst, angularity, EECs)	To-do	1 week	Dener
PHYS Benchmark[a]: Events - TEECs, NECs	NECs in progress (see here)	3 weeks	Derek (NECs)

* Assuming 50% FTE, including code review time

a) Desirable, but not required

b) Could do inclusive, HF-tagged, etc.

- **Notes:**
  - EM = “Electromagnetic”, H = “Hadronic
  - dRcst = constituent delta-R

Backup | Proposal: JANA2 2.4.3 Upgrade

- **PFA0 Status:** blocked until we upgrade to JANA2 2.4.3
  - But this is only needed to resolve bug exposed when using track-protocluster links
  - And PFA0 should be only place in PFAlpha that uses track-protocluster links

∴ We could approach development in a way that defers JANA2 2.4.3 dependency as long as possible

- **Proposal:**
  1) PFA0 work is paused until collaboration is ready for 2.4.3 upgrade (left note in PR)
  2) Output of track-cluster matcher is used as input to PFA1 and work continues downstream

☞ This way we're maximizing our use of time AND the modular design of PFAlpha

Backup | Proposal: EMCal vs. HCal Flagging

- A crucial point for PF (and eID) is being able to distinguish EMCal vs. HCal clusters
  - **Currently there is no easy way to do this**
  - Two possible approaches to fix this
    a) Using new datatypes with split EMCal, HCal cluster fields edm4eic#104
    b) Or use the Cluster::type field to flag EMCal vs. HCal, eg. edm4eic#122

- Latter can be done exclusively in EICrecon, former requires change to data model
  - ☞ Preference in group was for latter at last discussion

- **Proposal:**
  - Follow edm4eic#122 for now and use Cluster::type filled with system ID to flag EMCal vs. HCal clusters

☞ This allows development to proceed on PFA1(b) and PFA2 *without* a data model change
  - And is a minimal solution, so can be easily extended or phased out at a later date

Backup | Initial PFA Attempt

- **PFAlpha:** initial stab in EICrecon#1186 (now closed)
  - Initial implementation aimed for just a single algorithm
  - Initially even aimed to handle all 3 regions of central detector in one algorithm…

- **The gist:**
  1) Project tracks through calos
  2) Associate all calo clusters in cone of size R around track
  3) Sum all calo energy in cone and subtract expected track energy from sum
  4) Merge leftover clusters in cones of size R
  5) **Return PFObjects (reco. particles)**
     - Tracks
     - Subtracted, merged cluster

- **Clear Drawbacks!**
  - ☒ Monolithic by definition
  - ☒ Hard to maintain, evolve
  - ☒ Wiring in new PF algorithms means rewriting lots of code

Parameters
- R_{sum}^{ECal}: radius in (η, φ) in which to combine ECal clusters
- R_{sum}^{HCal}: same but for HCal
- f_{sub}^{ECal}: fraction of track energy to subtract from ECal clusters
- f_{sub}^{ECal}: same but for HCal

Backup | Mapping Initial PFA Attempt Onto Current

PFA-1

1) Subtract projected E_{trk} from ECal, HCal clusters
   a) Identify seed (highest p_{trk}) track projection at inner face of ECal
   b) Sum E_{trk} of all projections in R_{sum}^{ECal}, R_{sum}^{HCal} of seed
   c) Sum E_{clust} of all ECal, HCal clusters in R_{sum}^{ECal}, R_{sum}^{HCal} respectively
   d) If $\Sigma E_{trk}^{ECal,HCal} < \Sigma E_{clust}^{ECal,HCal}$
      i. Subtract $f_{trk}^{ECal,HCal} \times E_{trk}^{ECal,HCal}$ of nearest projection from each cluster
      ii. Pass subtracted clusters onto step 2
   e) Repeat 1(a) – 1(d)(ii) until all projections have been used

PFA0

PFA1(a)

PFA2

2) Combine remaining ECal, HCal clusters into topo-clusters
   a) Combine nearby ECal, HCal clusters
      i. Identify seed (highest E_{clust}) ECal cluster
      ii. Merge all ECal, HCal clusters in R_{sum}^{ECal}, R_{sum}^{HCal} of seed
      iii. Repeat 2(a)(i) – 2(a)(iii) until no ECal clusters are left
   b) Combine remaining HCal clusters
      i. Identify seed HCal cluster
      ii. Add all HCal clusters in R_{sum}^{HCal} of seed
      iii. Repeat 2(b)(i) – 2(b)(iii) until no HCal clusters left
3) Return PFObjects

PFA1(b)/PFA3

- **Note:** new approach *also* splits up PFA0 - 2 into separate calorimeters/eta regions