Data Model for PID at ePIC

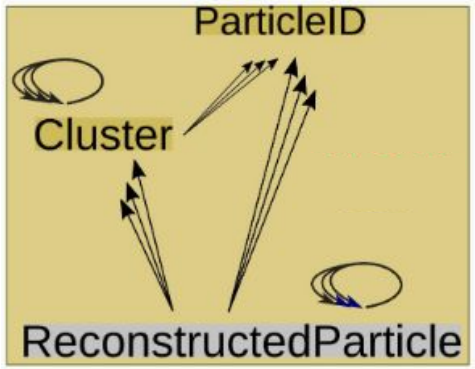
O. Hartbrich, D. Kalinkin

Charge for this talk

- our goal is to describe state of ePIC's EDM and how those fit into global PID
- the scope of this discussion is focused on **data needed for analysis of PID responses**
- additional detector-level information required for expert-level PID studies is out of scope

Event Data Model (EDM)

- ePIC software framework is based around careful **data structure** design co-developed with a set of *modular* “**algorithms**”
- Reusability is highly encouraged in design of both
- We start from structures defined by community-developed EDM4hep project
- EIC-specific **Data structures** are developed as EDM4eic




```
edm4hep::ParticleID:
  Description:  "ParticleID"
  Author: "EDM4hep authors"
  Members:
    - int32_t   type           // userdefined type
    - int32_t   PDG            // PDG code of this id - ( 999999 ) if unknown
    - int32_t   algorithmType  // type of the algorithm/module that created this hypothesis
    - float likelihood     // likelihood of this hypothesis - in a user defined normalization
  VectorMembers:
    - float parameters     // parameters associated with this hypothesis
  OneToOneRelations:
    - edm4hep::ReconstructedParticle particle // the particle from which this PID has been computed
```

State of PID structures in our EDM

- Two implementations that rely on edm4hep::ParticleID:
  - [1] [2] **PID LUTs** for *pfRICH*, *DIRC*, *TOF*, *DRICH*: e±, π±, K±, p±
  - [3] **ML on clusters** for *EEEMCal*: e-, π-

```
edm4eic::ReconstructedParticle:
  Description: "EIC Reconstructed Particle"
  Author: "W. Armstrong, S. Joosten, F. Gaede"
  Members:
    - int32_t           PDG              // PDG code for this particle
    // ... Definitions for energy, momentum, mass, ...
    - float             goodnessOfPID    // overall goodness of the PID on a scale of [0;1]
  OneToOneRelations:
    // ...
    - edm4hep::ParticleID  particleIDUsed // particle ID used for ~~the kinematics of~~ this particle
  OneToManyRelations:
    // ... Defitintions for relations to other clusters, track, particles ...
    - edm4hep::ParticleID  particleIDs    // All associated particle IDs for this particle (not so
```

Not yet

```
edm4hep::ParticleID:
  Description:  "ParticleID"
  Author: "EDM4hep authors"
  Members:
    - int32_t   type
    - int32_t   PDG
    - int32_t   algorithmType
    - float likelihood     //
```

```
edm4hep::ParticleID:
  Description:  "ParticleID"
  Author: "EDM4hep authors"
  Members:
    - int32_t   type
    - int32_t   PDG
    - int32_t   algorithmType
    - float likelihood     //
```

```
edm4hep::ParticleID:
  Description:  "ParticleID"
  Author: "EDM4hep authors"
  Members:
    - int32_t   type
    - int32_t   PDG
    - int32_t   algorithmType
    - float likelihood     //
```

```
...
```

  - EDM4hep insists on inverting the relation, i.e. ParticleID points to ReconstructedParticle [4] [5]
- The ML method currently “misuses” the **likelihood field and provide posterior P(hypot|meas)** instead of the **likelihood P(meas|hypot)**
- The LUTs also “misuse”, providing **distribution of model outputs conditioned by true class P(prediction|hypot)**. Main charge for LUTs was to fill the PDG field, reproducing a realistic misclassification distribution.

State of PID structures in our EDM

- One (the IRTv1) relies on edm4eic::CherenkovParticleIDHypothesis

```
edm4eic::CherenkovParticleID:
  Description: "Cherenkov detector PID"
  Author: "A. Kiselev, C. Chatterjee, C. Dilks"
  Members:
    - float             npe               // Overall photoelectron count
    - float             refractiveIndex   // Average refractive index at the Cherenkov photons' vertices
    - float             photonEnergy      // Average energy for these Cherenkov photons [GeV]
  VectorMembers:
    - edm4eic::CherenkovParticleIDHypothesis hypotheses         // Evaluated PDG hypotheses
    - edm4hep::Vector2f                      thetaPhiPhotons    // estimated (theta,phi) for each Cherenkov photon
  OneToOneRelations:
    - edm4eic::TrackSegment                  chargedParticle    // reconstructed charged particle
  OneToManyRelations:
    - edm4eic::MCRecoTrackerHitAssociation   rawHitAssociations // raw sensor hits, associated with MC hits
```

```
## PID hypothesis from Cherenkov detectors
edm4eic::CherenkovParticleIDHypothesis:
  Members:
    - int32_t           PDG             // PDG code
    - float             npe             // Overall photoelectron count
    - float             weight          // Weight of this hypothesis, such as likelihood, moment, etc.
```

- Rich (no pun intended) in terms of expert and non-expert information (comes with a quality estimate)

Combination of PID information I

- (Based on previous presentations by U. Tamponi and OH, e.g. at Lehigh collab meeting summer ‘24)
- Each PID-capable detector provides a likelihood value for each track for each of the distinct PID hypotheses
  - In Belle II: electron, muon, pion, kaon, proton, deuteron
  - Calculate likelihoods from comparing measured signal to expectation for hypothesis (Usually analytical models or MC or data templates)

$$\mathcal{L}_{\alpha}^{d} = \mathcal{L}^{d}(\mathbf{x}|\alpha)$$

Likelihood for hypothesis α from detector d that observed x hits

- This is stored in the PID structure of the EDM as described

Combination of PID information II

- Combining these individual detector likelihoods into a global likelihood is straightforward:

$$\mathcal{L}(\mathbf{x}|i) = \exp\left(\sum_{d}^{d\in D} \log\mathcal{L}^{d}(\mathbf{x}|i)\right)$$

Likelihood for hypothesis i from all detectors

  - If detector does not have acceptance for given track, set logL = 0 for all hypotheses
  - (n.b.: only true for fully uncorrelated measurements, ML combination can improve on hidden correlations)
- Bayesian inference converts these likelihoods into true PID probabilities:

$$P(A_i|\mathbf{x}) = \frac{P(\mathbf{x}|A_i)\cdot P(A_i)}{\sum_j P(\mathbf{x}|A_j)P(A_j)} \quad \Rightarrow P(i|\mathbf{x}) = \frac{\mathcal{L}_i}{\sum_j \mathcal{L}_j}$$

User Considerations (in Belle II)

- B2 only saves the LogL values in “reconstructed data” available for analysis
  - ~20% of raw data are always available for extra studies
- PID probabilities are calculated on-fly by the analysis libraries
  - Helper functions available to simplify user access:
  - Users can choose which type or probability (global, binary, ternary…)
  - Users can choose which detectors are to be used
- Expert variables (raw likelihoods) accessible, but “hidden”

```
electronID, muonID, pionID, kaonID, protonID, deuteronID
```
← likelihood ratio, e.g. logL(e)/sum_i(logL(i))

```
pidPairChargedBDTScore(pdgCodeHyp, pdgCodeTest)
```

“Expert” variables

```
pidLogLikelihoodValueExpert(pdgCode, detectorList)
pidDeltaLogLikelihoodValueExpert(pdgCode1, pdgCode2, detectorList)
pidPairProbabilityExpert(pdgCodeHyp, pdgCodeTest, detectorList)
pidProbabilityExpert(pdgCodeHyp, detectorList)
```

Conclusion/Summary

- The basic PID “traits” structures are already present in EDM4eic
- Reference algorithms for combination can be integrated into EDM4eic utility library
- Principled approach to global PID requires that start implementing proper likelihoods
- Need a list of quality estimate variables for each subsystem