Vertex EDM

Joe Osborn
Brookhaven National Laboratory
October 5, 2023

Vertex Data Type

```
## ============================================================================
## Vertexing
## ============================================================================

edm4eic::Vertex:
  Description: "EIC vertex"
  Author: "W. Armstrong, S. Joosten, based off EDM4hep"
  Members:
    - int32_t             primary       // Boolean flag, if vertex is the primary vertex of the event
    - float               chi2          // Chi-squared of the vertex fit
    - float               probability   // Probability of the vertex fit
    - edm4hep::Vector3f   position      // [mm] position of the vertex.
    ## this is named "covMatrix" in EDM4hep, renamed for consistency with the rest of edm4eic
    - edm4eic::Cov3f      positionError // Covariance matrix of the position
    - int32_t             algorithmType // Type code for the algorithm that has been used to create the vertex - check/set the colle
    ## Additional parameter not in EDM4hep: vertex time
    - float               time          // Vertex time
  VectorMembers:
    - float               parameters    // Additional parameters related to this vertex - check/set the collection parameter "Vertex
  OneToOneRelations:
    ## @TODO: why one and not multiple particles?
    - edm4eic::ReconstructedParticle associatedParticle // reconstructed particle associated to this vertex.
```

- Current vertex data type is not optimal
  - Missing time covariance, primary is not clear in streaming context (what is “the” PV?), should have relations to tracks, not particles, could use chi2/NDF…

Proposal

```
## ============================================================================
## Vertexing
## ============================================================================

edm4eic::Vertex:
  Description: "EIC vertex"
  Author: "J. Osborn"
  Members:
    - uint32_t               type       // Type flag, to identify what kind of vertex is identified
    - float               chi2          // Chi-squared of the vertex fit
    - float               ndf   // NDF of the vertex fit
    - edm4hep::Vector4f   fullPosition      // [mm] position + time t0 of the vertex.
    ## this is named "covMatrix" in EDM4hep, renamed for consistency with the rest of edm4eic
    - edm4eic::Cov4f      fullPositionError // Covariance matrix of the position
  OneToManyRelations:
    - edm4eic::ReconstructedTrack associatedTracks // reconstructed tracks associated to this vertex.
```

Acts::Vertex

```
private:
 Vector4 m_position = Vector4::Zero();
 SymMatrix4 m_covariance = SymMatrix4::Zero();
 std::vector<TrackAtVertex<input_track_t>> m_tracksAtVertex;
 double m_chiSquared = 0.;  // chi2 of the fit
 double m_numberDoF = 0.;   // number of degrees of freedom
```

- Acts vertex object is an example of a more flexible type
- We need OneToManyRelations for tracks to vertex
- I think we agreed this is what we would want for now. Other opinions?

To-Dos - Vertexing

```
edm4eic::MCVertex:
  Description: "MC EIC vertex"
  Author: "J. Osborn"
  Members:
    - uint32_t            type       // Type flag, to identify what kind of MC vertex is identified (e.g. background vs. primary)
    - edm4hep::Vector4f   position   // [mm] position + time t0 of the MC vertex
  OneToManyRelations:
    - edm4eic::MCParticle sim        // reference to the MC particles associated to the MCVertex
```

- Currently there is no MCVertex object. Propose we make one which will make vertexing evaluation more robust, especially in the presence of backgrounds

- This doesn’t need all of the detail a reco vertex needs (e.g. no covariance), so an object really only needs a 4 vector and some ID that indicates what kind of vertex (e.g. primary, secondary, what source of background, etc.)

- Write an algorithm that fills an MCVertex container with relevant truth vertices. Put the collection in PODIO output

To-Dos - Vertexing

- Alter vertexing algorithm to create new vertex data object
  - Caveat - this requires relating the vertex to reconstructed tracks
- We then need an algorithm to relate the reco vertices back to truth vertices
  - How is this currently done for the tracks? Full reco->truth hit tracing or something similar?