TruthVertexFinder

Due on 2015 - 2017

BILOKIN Sviatoslav [*]

PÖSCHL Roman, RICHARD François

LAL, Orsay, France

[*]bilokin@lal.in2p3.fr

Contents

Overview

TruthVertexFinder is a Marlin processor designed to extract secondary vertices from event generator collections.
Secondary vertex is a result of particle decay that does not coincide with main interation point in a detector. Vertices are produced by b- and c- hadrons are most interesting for physics analysis. The typical b-hadron decay has two vertices - secondary and ternary, while c-hadrons can have only one vertex per hadron. *TruthVertexFinder* can select secondary or ternary vertices and present them in an object-oriented way. It is designed for a vertex charge reconstruction study.
TruthVertexFinder has the following features:

- Tagging of the event by presence of quark pairs in event generetor collections - $t\bar{t}$, $b\bar{b}$ or $c\bar{c}$
- Searching for a quark by configured PDG type in generator collection and looking for particular hadron state after hadronization
- Looking for major particle decay chain by using a user-defined scheme.
- Constructing a vertex object for each major particle decay with corresponding position and other properties of a decay
- Adding prongs to a constructed vertex object. Prongs are generated particles that are visible in detector such as $e^{\pm}$, $\mu^{\pm}$, $\pi^{\pm}$ and others
- Creating of output collections for further usage
- Writing a *ROOT* file with information about vertices found

Important notice: TruthVertexFinder does not apply any selection cuts on particle kinematics.
There are two default major output collections:

1. *MCVertex* - is a collection of *EVENT::Vertex* objects. Each vertex object has the following properties:
   - Position - *getPosition()* method
   - Particle that created a vertex - *getAssociatedParticle()* method, this particle has its generated prongs in getParticles() method
2. *EGProngs* - collection of generated prongs with corresponding properties

All particles that are part of *MCVertex* collection are **EVENT::ReconstructedParticle* objects. These objects are copied from corresponding **EVENT::MCParticle* objects to match the *EVENT::Vertex* object architecture. It is recommended to use *EGProngs* collection if one has to work with *RecoMCTruthLink* collection.
The *EGProngs* collection has a set of parameters (*getParameters()* method) named *"trackIDs"*. These parameters are int numbers, the sign of each number indicates a parent quark/antiquark and digit stands for generation of vertex - secondary or ternary. For example value -2 means that this track is emerged from a secondary vertex of an antiquark, and value $+3$ - is for track from a ternary vertex of a quark. The *EGProngs* collection is a subset of *MCParticle* collection.

Run options

There should come an example steering file *example.xml* along with this document.
The central steering part of the algorithm is the user-defined decay chain scheme created by *DecayChainPDGs* parameter. This parameter has type of *vector<int>* and it represents a kind of particle PDG enumeration. Possible values are:

- 500 - B-mesons
- 400 - D-mesons
- 300 - K-mesons
- 5000 - b-baryons
- 4000 - c-baryons
- 5500 - b-hadrons
- 4400 - c-hadrons
- 15 - τ lepton
- 0 - any other particle

For example, the scheme "500 400 0" represents process $B \to D \to any$ and two vertex objects per found decay chain will be written by *TruthVertexFinder*. The 0 value is required to create a vertex for previous PDG type with any type of daughters.
Other parameters are:

- *tagPDG* - processor will tag the output collections by presence of pair of quarks with this PDG code in the event, possible values are: 4, 5, 6
- *initialQuarkPDG* - PDG code of initial quark for a decay chain, possible values are: 4, 5, 6
- *writeROOT* - parameter that regulates an output ROOT file, possible values are:
  - 0 - no ROOT output
  - 1 - basic ROOT output
  - 2 - extended ROOT file, for two vertex per quark **only**
- *ROOTFileName* - name of ROOT output file
- *Verbosity* - standard parameter, regulates text log output

Important notices: It is not recommended to set flavour of *initialQuarkPDG* different from flavour of hadrons in *DecayChainPDGs* parameter. Cases with more than two vertices per decay chain were not yet tested.

Description of the algorithm

The *TruthVertexFinder* has rich MVC-like structure of code subdivided into C++ classes. All classes are classified as follows:

- Processor - *TruthVertexFinder* launches the algoritm and writes down an output
- Operators
  - *MCOperator* - highly recursive algorithm provides a search for particle decays in *MCParticles* collections
  - *VertexMCOperator* - creates and handles *EVENT::Vertex* objects
  - *MathOperator* - scope of mathematical functions
- Domains
  - *MyVertex* - inherits *IMPL::VertexImpl* class
  - *DecayChain* - contains information about found decay chain
- Constrants - *ConstantStorage* stores constant values

Known drawbacks and problems

TruthVertexFinder has the following known restrictions:

- Color quark-gluon string problem - PYTHIA creates a quark-gluon string during hadronization step that fuses different quarks together. This fact forces to select corresponding hadrons by angle and momentum instead of using a parent/daughter relation. Therefore there is a small probability that a hadron will be related to a wrong quark if two hadrons are produced in the same direction.
- One or two decay chain restriction - the algoritm is designed to work on $e^+e^- \to q\bar{q}$ processes.
- Soft electron problem - sometimes in *MCParticles* collection there is a soft $e^\pm$ produced along with a D-meson by unknown reasons. This happens in less then 0.1% of events.
- Double D decays - the processes of type $B \to D\bar{D}$ have two ternary vertices and they are ignored by the algorithm.
- Decays of K_s with a low time of flight can contribute to a parent vertex, but at current stage of the program it is not handled.
- Extended mode of output ROOT files is only for two vertex per decay chain case.