Table of Contents

DaVinci Tutorial 6

This is all about running `DecayTreeTuple` and some more.

Slides

This tutorial corresponds to the slides last shown here. (i.e. that is an entire conference around the tutorials, you can read all of the slides, and they will take you though the tutorials step-by-step). Also see the talk on `DecayTreeTuple` here which is a nice introduction, but strays a little into DaVinciTutorial7 also.

Why am I looking at this?

We normally use ROOT nTuple structures to store just the information about data or simulation we really need to do our analyses. When done properly the ntuple will be much smaller than the corresponding full data DST.

This tutorial describes the tools which exist already in DaVinci to get event data or particle data into a ROOT nTuple structure.

Event or Particle? Simulation or Real Data?

The **DecayTreeTuple** framework comes in three basic flavours. Three basic algorithms which fill different kinds of nTuple.

- **DecayTreeTuple**: Fill information about particles, vertices, daughters, of a specific decay chain from a specific location. One entry per "head" particle (the mother decaying particle).
- **MCDecayTreeTuple**: Fill information about Monte Carlo truth particles, again of a specific decay chain, again one entry per "head" particle (the mother decaying particle).
- **EventTuple**: Fill information about the general properties of the event, how many tracks, run number/event number. One entry per event.

For each of these basic types there is a corresponding type of tool which fills certain information. There are many different copies of these tools to fill different information, and eventually get just the information you actually want in your ROOT file.

- **ParticleTupleTool**: Gets passed a particle and a tuple, fills information about the particle and returns. Only used in DecayTreeTuple.
- **MCParticleTupleTool**: Gets passed a Monte Carlo particle and a tuple, fills information about the Monte Carlo particle and returns. Basic ToolType for MCDecayTreeTuple, but can also be used in DecayTreeTuple, if carefully configured for the corresponding MC True particle (covered in tutorial 7).
- **EventTupleTool**: Doesn't get passed anything, because it fills general information about the event you are sitting in. Can be added to any of the three tuple types. If added to a DecayTreeTuple it fills identical information for each particle in the same event.

Common Tuple Tools

When you create a decay tree tuple, there are several variables that are automatically written into it:

- nCandidate
- totCandidates

- EventInSequence

Extra tuple tools can write particle, track, vertex, and event information into your ROOT tuple.

Let's concentrate on DecayTreeTuple for a bit, which fills information about particles, or the event those particles are in. Two tuple tools are automatically included in the tuple's tool list:

- TupleToolGeometry, which writes information related to the geometry of the vertex locations
- TupleToolKinematic, which writes kinematic variables

Some other common and useful tuple tools include:

- TupleToolPropertime, which simply writes the proper lifetime of reconstructed particles
- TupleToolAngles, which writes the decay angles of (charged) tracks
- TupleToolPid, which writes PID (particle identification) information for (charge) particles
- TupleToolTISTOS (TISTOS stands for "Trigger Independent of Signal or Trigger On Signal)
- TupleToolPrimaries, which writes the number and coordinates of primary vertices in the candidate events
- TupleToolTrackInfo, which writes track type and reconstruction information
- TupleToolEventInfo, which writes event information such as run number, magnet polarity, GPS time, bunch ID, etc.

Additionally, there are some tuple tools that provide MC information. Many of these are completely analogous to the above data TupleTools, but simply provide MC truth values for truth matched candidates (and appropriate null values for non-truth matched candidates). Others give other sorts of MC information. These include:

- TupleToolMCBackgroundInfo, which use some boolean variables to sort reconstructed candidates into some generic signal and background categories. See the tutorial on BackgroundCategory.

Data or MC

To avoid filling stuff in the Tuple that is not available, it's a good idea to put a switch in your options, or write smart configurables/classes to configure things for you.

```
simulation = False # or True, as you wish
```

Make two Tuples

Add an instance of `DecayTreeTuple` and add the appropriate tools:

```
from DecayTreeTuple.Configuration import *
from DaVinci4.solutions.Bs2JpsiPhi import SeqBs2JpsiPhi
# get the GaudiSequencer with everything we need
seq = SeqBs2JpsiPhi.sequence()
tuple = DecayTreeTuple()
tuple.Inputs = [ SeqBs2JpsiPhi.outputLocation() ]
# tuple.addTupleTool( "TupleToolGeometry") // already default
# tuple.addTupleTool( "TupleToolKinematic")// already default
tuple.addTupleTool( "TupleToolPropertime")
tuple.addTupleTool( "TupleToolPrimaries")
# tuple.addTupleTool( "TupleToolEventInfo")// already default
tuple.addTupleTool( "TupleToolTrackInfo")
tuple.addTupleTool( "TupleToolTagging")
if (simulation):
    tuple.addTupleTool( "TupleToolMCTruth")
    tuple.addTupleTool( "TupleToolMCBackgroundInfo")
tuple.Decay = "[B_s0 -> ^(J/psi(1S) -> ^mu+ ^mu-) ^(phi(1020) -> ^K+ ^K-)]CC"
```

`DecayTreeTuple` makes one entry per candidate and takes both `EventTupleTools` and `ParticleTupleTools`. The most difficult part of the exercise is to get the decay descriptor right. See the syntax here.

You can also add a Tuple with one entry per event. This is done using `EventTuple`.

```
etuple = EventTuple()
# etuple.addTupleTool("TupleToolEventInfo")// already default
if (simulation): etuple.addTupleTool("TupleToolGeneration")
etuple.addTupleTool("TupleToolTrigger")
```

An `EventTuple` can only take `EventTupleTools`. Since a candidate always also has an associated event, you can add the information from the event into every candidate. This is why EventTupleTools can equally well go into an EventTuple and/or a DecayTreeTuple.

`EventTupleTools` and `ParticleTupleTools` are tools that implement the `IEventTupleTool` and `IParticleTupleTool` tool interfaces, respectively. Look at the description of these interfaces in doxygen, starting from this page to get a full up to date list of all tools implementing these interfaces. For more information on the different tools and what they are for, you really should see the slides last shown here.

Job configuration

Run them outside of your sequence, and define an ntuple output file (which is different from the histogram file):

```
DaVinci().UserAlgorithms = [ tutorialseq, tuple, etuple ]
DaVinci().TupleFile = "DVNtuples.root"
DaVinci().Simulation = simulation
```

Next:

- Go to DaVinciTutorial7 if you want to know all about `DecayTreeTuple`.
- Go to DaVinciTutorialTips if you think you are ready for your analysis
- Go to DaVinciTutorial to pick between all the *Advanced* sections

-- PatrickKoppenburg - 01 Oct 2007 -- PatrickKoppenburg - 13 Jun 2008 -- PatrickKoppenburg - 05 Jan 2009 -- PatrickSKoppenburg - 16-Oct-2012 -- PatrickSKoppenburg - 30-Sep-2013