Subject: Re: PID package
Posted by Bertram Kopf on Fri, 07 Aug 2009 09:52:54 GMT
View Forum Message <> Reply to Message

Hi Stefano,

Stefano Spataro wrote
Not exactly. I think you have missed to read the minutes of the EVO meeting on 1st July 09, where the structure was proposed and somehow accepted:

http://panda-wiki.gsi.de/cgi-bin/view/Computing/Minutes01Jul2009

The correlator is just correlating the track to pid detectors, and create candidates for pid. This are charged candidates (at the moment called PndPidCandidate), and neutral candidate. The correlator is not doing PID at all, just correlation.

O.k. this means that the correlator should be only responsible for the collection of the properties of all subdetectors which might be useful for PID. When I look to your proposed data flow I come to the conclusion that it is only foreseen to collect all PID information, to apply a network etc. and based on this result to create a RhoCandidate. This means that just a part of the global PID will be supported. As you know most experiments make use of likelihood based algorithms where e.g. each detector provides PID likelihoods which will be combined afterwards. Does it mean that subdetector PID will be skipped completely? Abstract classes, mechanisms for the possibility to choose different algorithms at runtime and common tools to handle probabilities etc. are - I think - also very helpful to provide.
I don't find any of such classes and tools in the present code. Is there an idea and a concept to implement such things?

Stefano Spataro wrote

Quote:
c. an extrapolation of tracks to indiviual subdetectors (ToF, EMC, Mdt, DIRC, etc.) will be done and
d. detector specific PID related properties will be calculated there (e.g. truncated mean method for STT).
I think, this goes definitely far beyond that what should be done and should be provided in the PID related code. As you know the software has to be highly modular and flexible and has to make use of encapsulation in order to keep the code maintainable. This means that one has to decouple the different things which are currently done in the PndPidCorrelator and furthermore "non PID related" things should not be placed in the PID package at all.

Sorry but I have not understood at all this point.
Apart from correlation, dE/dx is calculated from the pid infos of the track. In theory my idea was to have single classes to retrieve useful informations from pid detectors. For complicated calculations like for cherenkov or tpc dE/dx this is my idea, but for simple calculations such as stt de/dx I have just implemented everything in the same class. Considering that (almost) nobody is working on pid detector informations, this makes life easier.

The track matching with the detectors STT, EMC, Tof and MVD are definitely done in this PndPidCorrelator. Such a track matching is in general detector dependent and not that easy and should be done separately. Does it mean that this is right now a workaround since essential parts in the reconstruction are missing? If so I am not sure if such a workaround - to put everything in one class/method - is the best solution. In my point of view life is easier to start with a proper design/structure and if something is still not there to provide dummy classes. With your workaround you have to change and reorder a lot of parts again and again.

Stefano Spataro wrote
The isntance is just a dummy function that is kept from old code, and that is never used. I could also removed it. PndPidCorrelator is a task, inheritated from FairTask and therefore from TTask, then all the basic functionality are there. It is not an object that everybody could use, but it has to be used inside our Run task list.

Then please remove the relevant lines immediately. Especially newcomers should not see such nonsense.

Stefano Spataro wrote
758 is already protected. In 763 and 764 the denominators are never zero.

yes, you are right!

Stefano Spataro wrote
Quote:
iii. in line 436 a stack overflow has not been caught.

Sorry but I have not understood this(the code was coming from stt developers). What is exactly the error?

sorry, I meant an overflow of the array. This must be caught!

Stefano Spataro wrote
For all the detectors the index of the corresponding hit TCA are kept, then it is always possible to do what you want.

Does it mean that this class provides just methods where one can ask for the index where to find the pointer in the branch? Does it mean that I have to take care of to build the transient object? This is very error prone because I have to know the correct tree and branch and in addition it is not type save. Isn't there a common mechanism available which is encapsulated from reconstruction?

Cheers,
Bertram.