NOvA-ART - Bug #7195

BackTracker::HitCollectionPurity and BackTracker::HitCollectionEfficiency are very slow

10/21/2014 01:08 PM - Brian Rebel

Status:	Rejected	**Start date:**	10/21/2014
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

These two methods together account for 1 second of real time per event when called from CosmicTrackAna_module.cc using Far Detector cosmic ray Monte Carlo. They should be examined for ways to be sped up.

History

#1 - 10/21/2014 01:21 PM - Christopher Backhouse

Are you calling these for every reconstructed object in the spill? I think that duplicates a lot of effort and you'd be better off with SlicesToMCTruthsTable(), which was introduced to combat exactly this sort of problem in CAFMaker. Still won't be blindingly fast, there's a lot of information to sort through, and quite a bit of work has been done in this area already, so likely no low-hanging fruit.

It may be a problem for you that that function takes PtrVector<Cluster>. Presumably you'd prefer vector<Cluster*>. I don't think there's any reason those particular need to be Ptrs, just the format CAFMaker had to hand.

#2 - 10/21/2014 01:27 PM - Brian Rebel

Hi Chris

yes, I am calling them for every muon in the event. I don't know what SlicesToMCTruthsTable() does or how to use it, can you provide an example?

Even so, we should find a way to improve the methods indicated in this issue because a naive user would just call them. A person should not have to be "in the know" to get around a problem, instead the problem should be solved.

#3 - 10/21/2014 01:32 PM - Christopher Backhouse

There's information on the function in doxygen. You give it a list of slices (or in your case tracks) and it gives back a list of the best "neutrino" (but it says it'll work for cosmic rays too) and the efficiency and purity for each of your inputs.

My memory is hazy, but I think the other methods wind up having to do almost as much work, but then only give you back the info for the reco object you asked for. Short of a significant amount of caching in BackTracker (which has already grown complicated) I'm not sure what can be done about that. Putting warnings on the functions in doxygen would probably be a good idea though.

#4 - 10/21/2014 01:36 PM - Brian Rebel

Thanks. I see some obvious problems in HitCollectionEfficiency that should improve things, like not making the HitToTrackIDE map multiple times and not copying the hits over and over again.

Why don't we we just take the faster algorithm from your method for finding purity and efficiency, stick that in the current methods and then have the SlicesToMCTruth call those? Then everyone would have faster algorithms.

#5 - 10/21/2014 01:44 PM - Christopher Backhouse

I only see us fetching the hit into a Ptr, which is a waste but not a big one. We do call HitToTrackIDE twice for each hit, which is once too many. I suspect that most of the time is manipulating the maps in those nested loops though.

If you can make these functions faster, we certainly should. Probably you should get sign-off from Susan that you didn't accidentally change the behaviour, she probably has the best idea of how these currently work.

I imagine the eff and pur functions do basically the same thing, and then just perform a different division at the end. If that's the case, we should consider merging them for a factor 2 speedup when you need both pieces of information.

#6 - 10/29/2014 09:38 AM - Susan Lein

If someone is identified to work on this, I can work with them to try to ensure that the end answers match the current answers. I don't have time to just do this on my own though.

#7 - 11/15/2016 04:55 PM - Alexander Himmel

- Status changed from New to Rejected