Subject: [SOLVED] Event filter after simulation phase
Posted by MG on Thu, 23 Apr 2020 12:57:06 GMT
View Forum Message <> Reply to Message

Hello!

I'm trying to implement a filter, that would run after the geant simulation and decide whether to store an event or not.

I've written a simple FairTask, which loads the branches I'm interested in and registers them for output. The exec method is just:

```
std::cout << "test1" << std::endl;

if(rand() % 2)
 FairRunSim::Instance()->MarkFill(false);
else
 FairRunSim::Instance()->MarkFill(true);
```

Now, if I use FairRunAna and run this as a task on already generated ntuple, it works flawlessly, saving about half of the events to the new ntuple.
However, if I use FairRunSim as above and add the task in my generation script, I can see that it runs after each event ("test1" is printed for each of the events), but all of the output is saved anyway. I've tried setting the storage of branches I register in the detector code to false, but it doesn't change anything. I've also tried changing the name under which I've registered one of the branches in the task (MCTrack -> MCTrack2) and it saved both MCTrack and MCTrack2 with the same number of events.

What am I doing wrong? Is it possible that the GeoTracks or MCTrack, which are also saved, but I can't control them from the detector (I think), force the tree write in between the simulation and my task?

Thank you very much for help!

Subject: Re: Event filter after simulation phase
Posted by Mohammad Al-Turany on Thu, 23 Apr 2020 13:11:36 GMT
View Forum Message <> Reply to Message

Hi MG,

MCTrack are controlled by the Stack filter if any (depend if you implement one). The TGeoTrack are only saved if you switch on the Track visualisation option.

The FairMCApplication forces the write of the entry in the tree (FinishEvent), so if you went to skip a whole event you have to process the tree after simulation.

best,

Mohammad

Subject: Re: Event filter after simulation phase
Posted by Florian Uhlig on Thu, 23 Apr 2020 14:02:04 GMT
View Forum Message <> Reply to Message

Hi,

please find attached an example task which does the work for you. As pointed out already by Mohammad the write during the simulation is steered by the FairMCApplication. In FairMCApplication ther is the function SetSaveCurrentEvent(Bool_t) which can be used to suppress the output of one event to file. The default is to write the event to file and this is reestablished after each event. So if you want to suppress the output you need a task which runs within the simulation which calls the function with kFALSE as parameter.

Ciao

Florian

File Attachments
1) CbmKresEtaMCAnalysis.cxx, downloaded 246 times
2) CbmKresEtaMCAnalysis.h, downloaded 250 times

Subject: Re: Event filter after simulation phase
Posted by MG on Thu, 23 Apr 2020 16:41:12 GMT
View Forum Message <> Reply to Message

Thank you very much!

Subject: Re: Event filter after simulation phase
Posted by MG on Fri, 24 Apr 2020 21:05:23 GMT
View Forum Message <> Reply to Message

I got a bit stuck again. I've implemented custom Stack based on the example, along with a simple filter on track momentum. What I've found is that the tracks are indeed removed, but hits from these tracks registered in detector are not, they just have their trackID set to -2. I would like to remove these hits as well.

I've tried updating the loop in FairStack::UpdateTrackIndex method, adding

```
if(-2 == point->GetTrackID())
  hitArray->RemoveAt(iPoint);
```

at the end, after the indices are set properly and also modified the loop to go backwards, so I don't break the indexing by removing items:

```
for (Int_t iPoint = nPoints - 1; iPoint >= 0; --iPoint)
```

This produces interesting results, which I've found after printing the size of hitArray before and after the call to RemoveAt. Basically, it works correctly up to some point and then stops removing items, even though the index is smaller than the size of the array.

Example:
Quote:
index 13 trackID -2
size before 14
size after 13
index 12 trackID -2
size before 13
size after 12
index 11 trackID -2
size before 12
size after 11
index 10 trackID 7
index 9 trackID -2
size before 11
size after 11
index 8 trackID -2
size before 11
size after 11

This behavior continues through many events and then it crashes with segmentation violation at some point (like 800 events in). It seems to always appear after the first case of trackID not being -2, as if only removing items from the end of CloneArray worked. My guess is that I'm breaking something with memory, because the branch is already registered and filled.

Is there a better way to do this?

Thanks!

Subject: Re: Event filter after simulation phase
Posted by Florian Uhlig on Fri, 24 Apr 2020 21:39:13 GMT
View Forum Message <> Reply to Message

I don't know what you want to do but the example I add only can switch off the writing of a complete event to file. You simply skip the step where you write to file and continue with the next event. It is not meant to partly write data from the event.

Subject: Re: Event filter after simulation phase
Posted by MG on Sun, 26 Apr 2020 17:38:10 GMT
View Forum Message <> Reply to Message

This issue is completely unrelated to the previous one. The event filter works great!

Sorry for being misleading, I was referring to the example of stack implementation in the FairRoot repository. I would like to be able to filter out tracks and remove hits from them.

Subject: Re: Event filter after simulation phase
Posted by Florian Uhlig on Mon, 27 Apr 2020 06:39:50 GMT
View Forum Message <> Reply to Message

Could you please create a new issue for the problem and mark the current one as solved. Simply change the title to

[SOLVED] Event filter after simulation phase
