PlotUtils Management - Task #23175

For merging data antuples, MergeTool should merge all runs for a playlist into a single file

08/27/2019 11:27 AM - Benjamin Messerly

Status:	New	**Start date:**	08/27/2019
Priority:	Normal	**Due date:**	
Assignee:	Benjamin Messerly	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour

Description

tl;dr load and loop times for data tuples merged into one file-per-run are MUCH worse compared to data tuples merged into one file-per-playlist. submitMergeToGrid.py and MergeTool currently do the latter for former. This should be fixed. Below is my summary of the issue and whether and why it should be fixed.

According to a comment in submitMergeToGrid.py, the intended behavior was to merge all data anatuples for a playlist into a single file. Instead right now, all subruns are merged together resulting in one file per run (which is indeed the desired behavior for data). Aaron B reportedly looked into this but wasn't able to solve it.

The only reason I'm bringing this up is because I'm seeing extreme speed differences between the two methods when looping over data.

We already know that merging is good and root's switching between files is a huge time suck, but I quantify that here with some quick time tests on data event selection.

Test details:

- I have my own merging code that I used to merge all the data runs for a playlist into a single file. I compare with the one-file-per-run method that the standard submitMergeToGrid outputs.
- ~98% of FHC POT, data, no systematics (obviously), just a standard loop with cuts and filling histograms.
- I'm using PU::ChainWrapper in both cases.
- Files are being stored in /pnfs/minerva/persistent and are NOT being accessed with xrootd.

One file per playlist trial 1: 1m44s
One file per playlist trial 2: 0m47s

One file per run trial 1: 39m:53s
One file per run trial 2: 14m16s

Conclusions:

- One-file-per-playlist is blisteringly fast in both cases.
- One-file-per-run, Trial 1 is terrible.
- In both cases, trials 1 and 2 were performed back-to-back. Trial 2 times in both cases are significantly lower, so there must be some kind of "loading into memory" going on with the pnfs access.
- Trial 2 times for one-file-per-run are more reasonable, but still! It's that first time that could throw you off for the whole day.

If you're running on the grid already, then you're probably (maybe?) insensitive to this, but IMO this is the difference between whether you need to run on the grid at all or not. Before I switched to the standard merging code, I got used to looping full data on the gpvm's in no time. When I switched and my times skyrocketed, I started looking into this.