Table of Contents

StreamingTaskForce

Introduction

When data is collected from the LHCb detector, the raw data will be transferred in quasi real time to the LHCb associated Tier 1 sites for the reconstruction to produce rDST files. The rDST files are used for stripping jobs where events are selected for physics analysis. Events selected in this way are written into DST files and distributed in identical copies to all the Tier 1 sites. These files are then accessible for physics analysis by individual collaborators. The stripping stage might be repeated several times a year with refined selection algorithms.

This report examines the needs and requirements of streaming at the data collection level as well as in the stripping process. We also look at how the information for the stripping should be made persistent and what bookkeeping information is required. Several use cases are analyzed for the development of a set of recommendations.

The work leading to this report is based on the streaming task force remit available as an appendix.

More background for the discussions leading to the recommendations in this report can be found in the Streaming Task Force Hypernews.

Definition of words

- A **stream** refers to the collection of events that are stored in the same physical file for a given run period. Not to be confused with I/O streams in a purely computing context (e.g. streaming of objects into a Root file).
- A **selection** is the output of a given selection during the stripping. There will be one or more **selections** in a given stream. It is expected that a selection should have a typical (large) size of 10^6 (10^7) events in 2 fb^{-1}. This means a reduction factor of 2 x 10^4 (10^3) compared to the 2 kHz input stream or an equivalent rate of 0.1 (1.0) Hz.

Use cases

A set of use cases to capture the requirements for the streaming were analyzed:

- Flavour tagging - Stripping of Flavor Tagging calibration data.
- A sparse analysis - Branching ratio of B_s $\mu^+\mu^-$.
- A high branching ratio analysis - CP violation in B^0 $D^{\pm}$.
- Detector calibration - Particle ID and alignment.
- Recovery - How to recover from a bug in parts of the stripping.
- Development - How to develop and test a new selection.
- Pilot run - Perform a detector calibration analysis with data from the 2007 pilot run.
- Complexity - Run an analysis that requires more than one one mode for the fit.

The analysis related to the individual use cases is documented in the Wiki pages related to the streaming task force.

Experience from other experiments

Other experiments with large data volumes have valuable experience. Below are two examples of what is

done elsewhere.

D0

In D0 the data from the detector has two streams. The first stream is of very low rate and selected in their L3 trigger. It is reconstructed more or less straight away and its use is similar to the tasks we will perform in the monitoring farm. The second stream contains all triggered data (including all of the first stream). Internally the stream is written to 4 files at any given time but there is no difference in the type of events going to each of them. The stream is buffered until the first stream has finished processing the run and updated the conditions. It is also checked that the new conditions have migrated to the remote centers and that they (by manual inspection) look reasonable. When the green light is given (typically in less than 24h) the reconstruction takes place at 4 remote sites (hence the 4 files above).

For analysis jobs there is a stripping procedure which selects events in the DST files but does not make copies of them. So an analysis will read something similar to our ETC files. This aspect is **not** working well. A huge load is experienced on the data servers due to large overheads in connection with reading sparse data.

Until now reprocessing of a specific type of physics data has not been done but a reprocessing of all B triggers is planned. This will require reading sparse events once from the stream with all the raw data from the detector.

BaBar

In BaBar there are a few different streams from the detector. A few for detector calibration like e^+e^- e^+e^- (Bhabha events) are prescaled to give the correct rate independent of luminosity. The dominant stream where nearly all physics come from is the hadronic stream. This large stream is not processed until the calibration constants are ready from the processing of the calibration streams for a given run.

BaBar initially operated with a system of rolling calibrations where calibrations for a given run *n* were used for the reconstruction of run *n+1*, using the so called 'AllEvents' stream. In this way the full statistics was available for the calibrations, there was no double processing of events but the conditions were always one run late. A consequence of this setup was that runs had to be processed sequentially, in chronological order, introducing scaling problems. The scaling problems were worsened by the fact that individual runs were processed on large farms of CPUs, and harvesting the calibration data, originating from the large number of jobs running in parallel, introduced a severe limit on the scalability of the processing farm. These limits on scalability were successfully removed by splitting the process of rolling calibrations from the processing of the data. Since the calibration only requires a very small fraction of the events recorded, these events could easily be separated by the trigger. Next this calibration stream is processed (in chronological order) as before, producing a rolling calibration. As the event rate is limited, scaling of this 'prompt calibration' pass is not a problem. Once the calibration constants for a given run have been determined in this way and have been propagated into a conditions database, the processing of the 'main stream' for that run is possible. Note that in this system the processing of the main physics data uses the calibrations constants obtained from the same run, and the processing of the 'main stream' is not restricted to a strict sequential, chronological order, but can be done for each run independently, on a collection of computing farms. This allows for easy scaling of the processing.

The reconstructed data is fed into a subsequent stripping job that writes out DST files. On the order of 100 files are written with some of them containing multiple selections. One of the streams contains **all** hadronic events. If a selection has either low priority or if its rejection rate is too poor an ETC file is written instead with pointers into the stream containing all hadronic events.

Data are stripped multiple times to reflect new and updated selections. Total reprocessing was frequent in the beginning but can now be years apart. It has only ever been done on the full hadronic sample.

Proposal

Here follows the recommendations of the task force.

Streams from detector

A single bulk stream should be written from the online farm. The advantage of this compared to a solution where several streams are written based on triggers is:

- Event duplication is in all cases avoided within a single subsequent selection. If a selection involves picking events from more than one detector stream there is no way to avoid duplication of events. To sort this out later in an analysis would be error prone.

The disadvantages are:

- It becomes harder to reprocess a smaller amount of the dataset according to the HLT selections (it might involve sparse reading). Experience from past experiments shows that this rarely happens.
- It is not possible to give special priority to a specific high priority analysis with a narrow exclusive trigger. As nearly every analysis will rely on larger selections for their result (normalization to *J/* signal, flavor tagging calibration) this seems in any case an unlikely scenario.

With more exclusive HLT selections later in the lifetime of LHCb the arguments might change and could at that point force a rethink.

Many experiments use a **hot** stream for providing calibration and monitoring of the detector as described in the sections on how streams are treated in BaBar and D0. In LHCb this should be completely covered within the monitoring farm. To be able to debug problems with alignment and calibration performed in the monitoring farm a facility should be developed to persist the events used for this task. These events would effectively be a second very low rate stream. The events would only be useful for debugging the behavior of tasks carried out in the monitoring farm.

Processing timing

To avoid a backlog it is required that the time between when data is collected and reconstructed is kept to a minimum. As the first stripping will take place at the same time this means that all calibration required for this has to be done in the monitoring farm. It is advisable to delay the processing for a short period (8 hours?) allowing shifters to give a green light for reconstruction. If problems are discovered a run will be marked as bad and the reconstruction postponed or abandoned.

Number of streams in stripping

Considering the low level of overlap between different selections, as documented in the page the appendix on correlations, it is a clear recommendation that we group selections into a small number of streams. This has some clear advantages compared to a single stream:

- Limited sparse reading of files. All selections will make up 10% or more of a given file.
- No need to use ETC files as part of the stripping. This will make data management on the Grid much easier (no need to know the location of files pointed to as well).
- There are no overheads associated with sparse data access. Currently there are large I/O overheads in reading single events (32kB per TES container), but also large CPU overheads when Root opens a file (reading of dictionaries etc.). This latter problem is being addressed by the ROOT team, with the introduction of a flag to disable reading of the streaming information.

The disadvantages are very limited:

- An analysis might cover more than one stream making it harder to deal with double counting of events. Lets take the B_s $\mu^+\mu^-$ analysis as an example. The signal will come from the two-body stream while the BR normalization will come from the $J/$ stream. In this case the double counting doesn't matter though so the objection is not real. If the signal itself is extracted from more than one stream there is a design error in the stripping for that analysis.
- Data will be duplicated. According to the analysis based on the DC04 TDR selections the duplication will be very limited. If we are limited in available disk space we should reconsider the mirroring of all stripped data to all T1's instead (making all data available at 5 out of 6 sites will save 17% disk space).

The appendix on correlations shows that it will be fairly easy to divide the data into streams. The full correlation table can be created automatically followed by a manual grouping based mainly on the correlations but also on analyses that naturally belong together. No given selection should form less than 10% of a stream to avoid too sparse reading.

In total one might expect around 30 streams from the stripping, each with around 10^7 events in 2 fb^{-1} of integrated luminosity. This can be broken down as:

- Around 20 physics analysis streams of 10^7 events each. There will most likely be significant variation in size between the individual streams.
- Random events that will be used for developing new selections. To get reasonable statistics for a selection with a reduction factor of 10^5 a sample of 10^7 events will be required. This will make it equivalent to a single large selection.
- A stream for understanding the trigger. This stream is likely to have a large overlap with the physics streams but for efficient trigger studies this can't be avoided.
- A few streams for detailed calibration of alignment, tracking and particle identification.
- A stream with random triggers after L0 to allow for the development of new code in the HLT. As a narrow exclusive HLT trigger might have a rejection factor of 10^5 (corresponding to 10 Hz) a sample of 10^7 is again a reasonable size.

Monte Carlo data

Data from inclusive and "cocktail" simulations will pass through the stripping process as well. To avoid complicating the system is recommended to process these events in the same way as the data. While this will produce some selections that are irrelevant for the simulation sample being processed, the management overheads involved in doing anything else will be excessive.

Meta data in relation to selection and stripping

As outlined in the use cases every analysis requires additional information about what is analyzed apart from the information in the events themselves.

Bookkeeping information required

From a database with the meta data from the stripping is should be possible to:

- Get a list of the exact files that went into a given selection. This might not translate directly into *runs* as a given run will have its rDST data spread across several files and a problem could be present with just one of them.
- For an arbitrary list of files that went into a selection obtain some *B counting* numbers that can be used for normalizing branching ratios. This number might be calculated during the stripping phase.

- To correct the above numbers when a given file turns unreadable (i.e. should know exactly which runs contributed to a given file).
- When the stripping was performed to be able to recover the exact conditions used during the stripping.

It is **urgent** to start a review of exactly what extra information is required for this type of bookkeeping information as well as how the information is accessed from the command line, from Ganga, from within a Gaudi job etc. A working solution for this should be in place for the first data.

Information required in Conditions database

The following information is required from the conditions database during the analysis phase.

Trigger conditions for any event should be stored. Preferably this should be in the form of a simple identifier to a set of trigger conditions. What the identifier corresponds to will be stored in CVS. An identifier should never be re-used in later releases for a different set of trigger conditions to avoid confusion.

Identification of good and bad runs. The definition of bad might need to be more fine grained as some analysis will be able to cope with specific problems (like *no RICH* info). This information belongs in the Conditions database rather than in the bookkeeping as the classification of good and bad might change at a time after the stripping has taken place. Also it might be required to identify which runs were classified as good at some time in the past to judge if some past analysis was affected by what was later identified as bad data. When selecting data for an analysis this information should be available thus putting a requirement on the bookkeeping system to be able to interrogate the conditions.

Procedure for including selections in the stripping

The note LHCb-2004-031 describes the (somewhat obsolete) guidelines to follow when providing a new selection and there are released Python tools that check these guidelines. However, the experience with organizing stripping jobs is poor: for DC04 only 3 out of 29 preselections were compliant in the tests and for DC06 it is a long battle to obtain a stripping job with sufficient reduction and with fast enough execution time. To ease the organization:

- Tools should be provided that automate the subscription of a selection to the stripping.
- The actual cuts applied in the selections should be considered as the responsibility of the physics WGs.
- We suggest the nomination of stripping coordinators in each WG. They are likely to be the same person as the "standard particles" coordinators.
- If a subscribed selection fails automatic tests for a new round of stripping it is unsubscribed and a notification sent to the coordinator.