SL feedback for the computing TDR - 31/03/16

Black: Questions, Blue: Answers (feel free to edit them yourself). Right click and do ‘suggest edit’ if you would like to make comments.

1) Data format in between DST and muDST. Although several analysis will never be happy with muDST, it is also true that
having all tracks and calo clusters is, in some (probably many) cases, not needed. As an example, often isolation variables are calculated within a cone
and what lies outside of this cone is not used at all and could be ignored. We would like your feedback on what is the minimal set of information you need for analysis that currently are on DST to understand whether some reduction in event size can be achieved. In particular, coming back to the "cone", we would like to have some idea of how this cone can be defined and a quantitative estimate of the "size" of it. Any other idea to save space is of course welcome.

Almost all semileptonic analyses require part of the rest of the event to complete the analysis, so a pure muDST is likely to never be a viable option. In short, we believe that a big enough Lorentz invariant cone could be enough to preserve all the physics we would like to do. More details are below.

Isolation is a very big part of the selection framework, both neutral and charged. So one would need a neutral cone as well as a charged cone to have full functionality. This cone is sometimes very large for the neutral case, see for example https://indico.cern.ch/event/503448/session/1/contribution/7/attachments/1237303/1817385/20160302_DsstarFits.pdf where the radius is set to 0.4. Using a Lorentz invariant cone instead could make this tighter.

Another issue is understanding background. It is not enough to calculate some simple quantities of a cone around the candidate, often a specific particle is reconstructed to determine the background yield from data. The previous presentation linked above shows this, where a photon is added to a ground state Ds in order to control the background from excited Ds states. Another example is in LHCb-ANA-2014-048, where additional pions, kaons and even Ks mesons are reconstructed to control backgrounds. For the later, just saving all long tracks would not be enough. Again, for this use case, a loose enough Lorentz invariant cone could catch all possible cases.

2) Turbo will be the default in the upgrade (as is muDST in the Stripping now). How many analysis in your WG can not be moved to turbo? Keep in mind that point 1 and 2 are not orthogonal. We could envisage that in the future turbo will be able to provide different data formats in output.

Most of our analyses are triggered via the topological triggers. So the only way to put SL on turbo would be to put the entire topo lines on turbo which would require agreement from many WGs. However, if the SL group would be open to this idea, as long as the persist reco flag is enabled to allow us to reconstruct our signal + partially reconstructed backgrounds.

Is the topo + persist reco too expensive in the upgrade? If so, then we believe that a Lorentz invariant cone could also be viable, in line with above.

One possible solution we discussed was to write a set of more exclusive trigger lines which take as input the topological trigger. It would basically put the stripping online rather than offline. Given the timing constraints of the HLT, we are not sure if this is feasible.

3) Centralised ntuple production. Ntuples will never die and right now all users have to make them by running over the stripping output. Would you be happy if ntuples could be produced every couple of weeks centrally by the production team?
The idea is to have analysis "trains" leaving at fixed time intervals and each "wagon" is an ntuple. The WGs have to provided the code to generate the ntuples and ask it to be added to the next "train". Ntuples can be downloaded once ready from the bookkeeping for offline analysis. The main advantage is that the resources of the users can be moved to the production team which could optimise the resource usage such as data replication, number of jobs on the grid, etc...
If this scheme is adopted, in the future there will be a need for a dedicated liaison for each WG (similar to the MC and Stripping one in terms of workload).

When we discussed this in the WG there was a bit of confusion here - isn’t the stripping going to be removed for the upgrade? So wouldn’t this ntuple production be running on turbo candidates as opposed to stripping candidates? Is this just for data or also for simulation too?

If this is just for data, then it could be made to work and we would much prefer this to the stripping. One would have to be careful regarding new users to LHCb who are not familiar of which information to keep. One should also definitely still be able to run over a small number of events locally, look at a tuple, check everything makes sense before getting on the ‘train’. Presumably this can be set up quite trivially like the stripping test sample is currently.

For simulation this could be more of an issue, as one often needs to run over many different samples with different selections which tends to require to tries to get right. If one had to wait 2 weeks just to check a background via MC this could become a significant delay.

One technical issue is private c++ code, how would that be dealt with? Many people quickly write a tupletool to access a bit of information and do not want that to go through the full release process.

4) Event index. We want to understand if random event access is a viable option for the upgrade. The idea is that events are indexed according to trigger/stripping line they fired and everything is put into a sort of database. The users will make a query to the database which will return a list of events. Therefore users will read only the events they are interested in (I remind you that right now, one has to loop sequentially over a file, even if most of the events are not useful for the analysis) Most importantly, these events can be anywhere in the grid. The gain from this approach is a much faster access to the events and allows an optimisation of data replication. We would need input from the WG to unserstand

what information should go into the index (e.g. Stripping lines, Trigger lines, multiplicities etc)

If the stripping ceases to exist then it’s difficult to know how one would tag our events given that they all come from the topo which is a large fraction of the data. Not sure how this could help access in the SL WG. If there is some kind of stripping which could tag when we have a D0 for example, then that could be much more efficient.