Table of Contents

Stripping tips for Liasons and Stripping authors

General structure

The LHCbStripping is the procedure used to pass from raw data, which should never been accessed by analysts, to analysis datasets. The data selected by the Stripping procedure can be saved into two different formats: DST or MicroDST. DST contains all the information from the raw events, plus the candidates produced during the Stripping run. New candidates may be produced offline, tracks and events can be refitted and detector studies can rely on the raw information as hits... MicroDST are smaller files which only contain information written during the Stripping. This is useful for analysis datasets which only need to be refined offline, but it is impossible to rerun combinatorics or to access detector information.

Retention

The amount of data which is saved by the Stripping is called retention. Due to physical disk-space limitations, a global retention of 5% is allowed. Give the huge number of Stripping algorithms (called Stripping Lines) the average retention for each Stripping Line is of 0.05%. This is an indicative number, WGs can have Stripping lines with higher retention, but obviously the less, the better.

CPU time

Another important parameter is the processing time per event. It should be less than 0.5 ms per event per stripping line. To reduce CPU time, there are few tricks:

- cuts can be applied to daughter particles used as input for more combination by using FilterDesktop instead of repeating the cut as DaughersCuts in each CombineParticle algorithm
- cuts to a particle array should be applied as CombinationCuts, and as MotherCuts only if the event fit with vertex constraint is really required. It could be worth to apply loose cuts on kinematic variables (as P, PT and Masses) as CombinationCuts (using Array LoKi Functors e.g. APT, AM ...), and to refine them as MotherCuts (using particle functors e.g. PT, M ...)
- Sorting the requiredSelection array, input of CombineParticles algorithm, from the rarest daughter particle to the most common. In this way, the algorithm first checks for the rarest particles and if it doesn't find any, it avoid wasting time looking for the others. This can reduce drastically the CPU time.
- Using StdLoose particles whenever it is possible. Usually one needs to use harder criteria than those applied in StdLoose combinations. In this case it is preferable to use the FilterDesktop enviornment instead of CombineParticles, to avoid doubling the fit time. To cut on daughters (for example if you want to cut on muons PT from StdJpsi2MuMu) you can use the LoKi functors MAXTREE/MINTREE.

Streams and Stripping versions Stripping Lines are collected in Stripping Streams. There are Leptonic, Dimuon, Bhadron, Charm, Calibration ... streams. If the stream name contains the "COMPLETEEVENT" keyword, the stream writes to DST, usually the analogue (without such a keyword) writes to MicroDST.

Streams are collected in Stripping versions. The naming convention for Stripping lines is updated here.

Packages

Stripping software is composed of two packages:

StrippingSelections

Phys/StrippingSelections contains the GaudiPython configuration files to apply Stripping-level cuts. More than one Stripping Line can be contained in a python file. Python files are, as usual, in `Phys/StrippingSelections/python/StrippingSelection/`

To find the python code related to a given Stripping Line you can try using `grep your_line_name *.py`

Some line uses a more complex and general definition and it may be difficult to find the exact cuts applied in a line. In these cases it hopefully exists an exhaustive documentation of the Stripping Lines. Some reference:

- *beauty2charm* lines: Mike's presentation -- part 1 Mike's presentation -- part 2

Each python file defining Stripping lines should contain a `configuration dictionary`, a python dictionary with the default thresholds of applied cuts. The threshold names should indicate the cut they refer to as precisely as possible.

The configuration dictionary is passed to the LineBuilder (the python machinery to transform a set of cuts in a Stripping Line) at the line initialization. To tell the builder which threshold variables to expect in the argument python dictionary, the `__configuration_keys__` should contain the list of threshold variables in the configuration dictionary.

Even if it is a python package, `StrippingSelections` needs compilation. Compilation generates compiled python with `.pyc` extenstion. To compile, SetupDaVinci head, move to the `Phys/StrippingSelections/cmt` directory and run `cmt make`.

Test before committing

* Run "cmt TestPackage" in "cmt" directory of StrippingSelections * Run "python test_allstreams_instantiation.py" in "tests"

StrippingSettings (Coordinators or Liasons only)

The package `Phys/StrippingSettings` contains a database with all the threshold variables for each StrippingSelections python file. Once a line has been added to Phys/StrippingSettings, its configuration dictionary with default cuts, has to be copied in the StrippingSettings database. To do this follow these steps:

- change dir to `Phys/StrippingSettings/python/StrippingSettings` folder.
- choose the stripping version to upgrade (usually the latest one) and move to its folder, herein we'll assume we upgrade `stripping20`
- append to the python file corresponding to the WG responsible for the line the new configuration dictionary
- change dir to `Phys/StrippingSettings/cmt` and compile with `cmt make` (DaVinci project has to be set up)
- change dir to `Phys/StrippingSettings/python/StrippingSettings` and run `python makeDB.py Stripping20`, this creates the upgraded database and saves it as stripping.tmp
- overwrite the existing database by doing `mv stripping.tmp ../../dbase/stripping20`

More details are available in the StrippingSettings page.

Testing Stripping lines

Refer to the StrippingFAQ to test a single line.

To test the lines for a given Working Group (for example BandQ) you can use the following statements:

```
from StrippingSettings.Utils import strippingConfiguration
from StrippingSelections.Utils import buildStreams
from StrippingConf.Configuration import StrippingConf

config = strippingConfiguration('stripping20')
streams = buildStreams(config, WGs = ['BandQ'])

sc = StrippingConf( Streams = streams,
                    MaxCandidates = 2000,
                    AcceptBadEvents = False,
                    BadEventSelection = filterBadEvents,
                    TESPrefix = 'Strip'
                    )
from Configurables import StrippingReport
sr = StrippingReport(Selections = sc.selections())

DaVinci().appendToMainSequence( [ sc.sequence() ] )
DaVinci().appendToMainSequence( [ sr ] )
```

refer to `Phys/StrippinsSelections/tests/TestMyStrippingLineOn2012Data.py` to see how to setup the rest of the Stripping environment.

-- LucioAnderlini - 03-Sep-2012

This topic: LHCb > LHCbStrippingTipsForLiasons
Topic revision: r5 - 2013-03-26 - LucioAnderlini