Debugging DaVinci jobs

- Make sense of the log file
  - Timing Table
  - Statistics
- Add you own debugging
  - Decay Trees
  - Plots

June 2009 Tutorial

Patrick Koppenburg
Imperial College
London

Debugging

Suppose you have a selection based on `CombineParticles` and/or `FilterDesktop` that suddenly does not select anything, or enough, or too much, or not what you want.

Usually then people

- Panic
- Send a mail to lhcb-davinci@cern.ch (OK)
- Send a mail to Juan
- Send a mail to me (that's bad)

P.S. I don't talk about debugging C++. That would be a nice title for another talk.

Debugging

Suppose you have a selection based on `CombineParticles` and/or `FilterDesktop` that suddenly does not select anything, or enough, or too much, or not what you want.

No need to panic. There's plenty of information available you may not know about.

- `gaudirun.py -v`
- Efficiencies
- Plots
- Correlations
- MC-truth tools

Debugging workflow

The main difficulty is to identify what went wrong.

The symptom may be empty Tuples or histograms, an algorithm not being run . . .

1. Identify from when the problem occurs
2. Look at the timing table to know which algorithms ran and which didn't
3. Look at the statistics of the last algorithm that ran properly

➜ Let's see some examples

Quick Debugging

When an algorithm does not do what I expect I usually do the following :

1. **Look at timing table**

Timing Auditor (Gaudi)

The problem: `PreselBd2KstarMuMu` does not find any events. It gets input from `DimuonForPreselBd2KstarMuMu`, which uses data from `SelectTrueBd2MuMuKstar`.

The timing table says:

TIMER.TIMER	INFO	DaVinciMainSeq	12.500	15.810	7.490	80.7	20	0.316
TIMER.TIMER	INFO	DaVinci	0.000	0.004	0.003	0.0	20	0.000
TIMER.TIMER	INFO	SeqPreselBd2KstarMuMu	2.500	2.938	0.225	50.6	20	0.059
TIMER.TIMER	INFO	DimuonForPreselBd2KstarMuMu	0.000	0.020	0.003	0.3	20	0.000
TIMER.TIMER	INFO	PreselBd2KstarMuMu	0.000	0.000	0.000	0.0	0	0.000
TIMER.TIMER	INFO	SeqTruthFilter	697.500	708.206	196.718	2259.9	20	14.164
TIMER.TIMER	INFO	SelectTrueBd2MuMuKstar	690.500	701.014	189.245	2255.1	20	14.020
TIMER.TIMER	INFO	AllTracksBd2MuMuKstar	1.000	0.408	0.038	2.4	20	0.008
TIMER.TIMER	INFO	FoundAll	3.684	3.871	0.227	66.9	19	0.074

1. `PreselBd2KstarMuMu` never runs because `DimuonForPreselBd2KstarMuMu` never finds anything (`m_filterPassed = false`).
2. No surprise: `SelectTrueBd2MuMuKstar` is run after it!

Timing Auditor (Gaudi)

The problem: `PreselBd2KstarMuMu` does not find any events. It gets input from `DimuonForPreselBd2KstarMuMu`, which uses data from `Selec`

The ti

```
TIMER.TI                                                    0.316 |
TIMER.TI                                                    0.000 |
TIMER.TI                                                    0.059 |
TIMER.TI                                                    0.000 |
TIMER.TI                                                    0.000 |
TIMER.TI                                                   14.164 |
TIMER.TI                                                   14.020 |
TIMER.TI                                                    0.008 |
TIMER.TI                                                    0.074 |
```

1. F `For-`
   F `ed =`
   f
2. No

The Timing Auditor:

- Gives you the list of all sequences in the order they have been run,
- The number of times each algorithm has been run.
- It is on by default in **DaVinci**.

I **always** look first at the timing table.

Quick Debugging

When an algorithm does not do what I expect I usually do the following :

1. **Look at timing table**
2. **Look at `CombineParticles` statistics**

Finalisation Statistics

`DVAlgorithm` gives us the statistics of accepted events:

```
StdLooseKaons SUCCESS Number of counters : 1
 |    Counter |     #     |    sum     | mean/effˆ* | rms/errˆ*  |     min     |
 |*"#accept"  |       472 |        472 |(  100.000 +- 0.211864 )%|   -------   |
```

The `finalize()` method of `CombineParticles` would have told us something similar, with more details:

```
Jpsi2MuMu                                                    SUCCESS Number of counters : 5
 |    Counter                 |  #   |  sum | mean/effˆ* | rms/errˆ*  |     min     |     max
 | "# J/psi(1S) ->  mu+ mu- " | 1000 |  532 |    0.53200 |    0.52629 |      0.0000 |       2
 | "# mu+"                    | 1000 | 1242 |     1.2420 |     1.2072 |      0.0000 |      17
 | "# mu-"                    | 1000 | 1290 |     1.2900 |     1.1198 |      0.0000 |       9
 | "# selected"               | 1000 |  532 |    0.53200 |    0.52629 |      0.0000 |       2
 |*"#accept"                  | 1000 |  518 |(  51.8000 +- 1.58011  )%|   -------   |    -----
```

Look at the stats of the algorithms you are interested in. They will tell you if the algorithms did anything sensible.

Quick Debugging

When an algorithm does not do what I expect I usually do the following :

1. **Look at timing table**
2. **Look at `CombineParticles` statistics**
3. **Counters**

Counters

Many messages that used to be `WARNING` s are now just counted silently . . .

. . . but printed out at the end of the job

```
PrintBu2LLK.PhysDesktop                                          SUCCESS Number
 |    Counter                                          |     #      |    sum
 | "Empty primary vertex container at Rec/Vertex/Pr|         4 |
 | "No particles at location Phys/Presel09Bu2LLK/Pa|       587 |        58
 | "No relations table exists at Presel09Bu2LLK/Par|      1000 |       100
```

- Don’t worry about relations
- The ”No particles” message tells you how many times no `Particles` have been found at the given `InputLocations`
  - 587 times nothing
  - ➜ 413 times something

Counters

DaVinciMainSequence	2.515	5.610	0.541	1907.9	1000	5.610
DstWriters	0.001	0.002	0.001	0.0	1000	0.002
Sel09Bu2LLKFilterSequence	2.277	5.142	0.468	1907.7	1000	5.142
Bu2LLK09_DiLepton	0.186	0.193	0.062	7.4	1000	0.193
Presel09Bu2LLK	0.199	0.179	0.074	5.0	413	0.074
Sel09Bu2LLK	0.067	0.108	0.080	0.4	30	0.003
BTagging	0.167	0.389	0.001	274.9	1000	0.389
MonitoringSequence	0.221	0.350	0.071	267.3	1000	0.350
ExampleSeq	0.217	0.347	0.068	267.3	1000	0.347
PrintBu2LLK	0.139	0.274	0.002	267.2	1000	0.274

- Don’t worry about relations
- The ”No particles” message tells you how many times no `Particles` have been found at the given `InputLocations`
  - 587 times nothing
  - ➜ 413 times something

Quick Debugging

When an algorithm does not do what I expect I usually do the following :

1. **Look at timing table**
2. **Look at `CombineParticles` statistics**
3. **Counters**
4. **Use the Debug Tool**

The Decay Tree Tool

- The Decay Tree Tool provides a human-readable dump of the event
- It works both with MC truth and with reconstructed particles
- It produces the following output:

```
<----------------------------------------- Particle -------------
       Name          E          M          P          Px         Py          Pz
                    GeV        GeV        GeV        GeV        GeV         GeV
B_s0              255.062      8.686    254.915    -20.824     -0.062     254.063
+-->J/psi(1S)     202.675      3.127    202.651    -19.344     -1.318     201.721
|+-->mu+           91.705      0.106     91.705     -7.480     -1.478      91.388
|+-->mu-          110.970      0.106    110.970    -11.865      0.160     110.334
+-->phi(1020)     52.387      1.030     52.377     -1.479      1.256      52.341
 +-->K-           21.810      0.494     21.804     -0.498      0.523      21.792
 +-->K+           30.577      0.494     30.573     -0.981      0.733      30.549
```

Using the Decay Tree Tool

The Decay Tree Tool can be used directly from an algorithm, for instance to print only when something goes wrong. It is not already interfaced in `DVAlgorithm`.

- Declare it:

```
#include "Kernel/IPrintDecayTreeTool.h"
```

- Use it:

```
IDecayTree Tool* m_debug =  tool<IPrintDecayTreeTool>( "PrintDecayTree
m_debug->printTree(part [, depth]);
m_debug->printAncestor(mcpart);
```

- Configure it:

```
jpsi2mumu.addTool(PrintDecayTreeTool())
jpsi2mumu.PrintDecayTreeTool.Information = "Name E M P Px Py Pz Pt phi
jpsi2mumu.PrintDecayTreeTool.PrintDepth = 3
```

- There are other methods and options. Have a look at the Decay Tree Tool Doxygen.

Decay Tree Algorithms

There are algorithms provided that call the Decay Tree Tool:

`DumpEvent`: No options. Dumps the whole MC event.

```
tutseq.Members += [ DumpEvent() ]
```

`PrintMCTree`: Prints the MC decay tree of particles of a given PID

```
from Configurables import PrintMCTree, PrintMCDecayTreeTool
mctree = PrintMCTree("PrintTrueBs")
mctree.addTool( PrintMCDecayTreeTool() )
mctree.PrintMCDecayTreeTool.Information = "Name M P Px Py Pz Pt chi2"
mctree.ParticleNames = [  "B_s0", "B_s~0" ]
mctree.Depth = 2  # down to the K and mu
tutorialseq.Members += [ mctree ]
```

Decay Tree Algorithms

`PrintDecayTree`: Prints the reconstructed tree

```
from Configurables import PrintDecayTree, PrintDecayTreeTool
tree = PrintDecayTree("PrintFoundBs")
tree.InputLocations = [ "Bs2JpsiPhi" ]
tree.addTool( PrintDecayTreeTool() )
tree.PrintDecayTreeTool.Information = "Name M P Px Py Pz Pt chi2"
tree.PrintTruth = True
tutorialseq.Members += [ tree ]
```

I use these algorithms a lot as they can help to check if the decay you look for is actually there without having to configure the Decay Finder.

Quick Debugging

When an algorithm does not do what I expect I usually do the following :

1. **Look at timing table**
2. **Look at `CombineParticles` statistics**
3. **Counters**
4. **Use the Debug Tool**
5. **Look at plots**

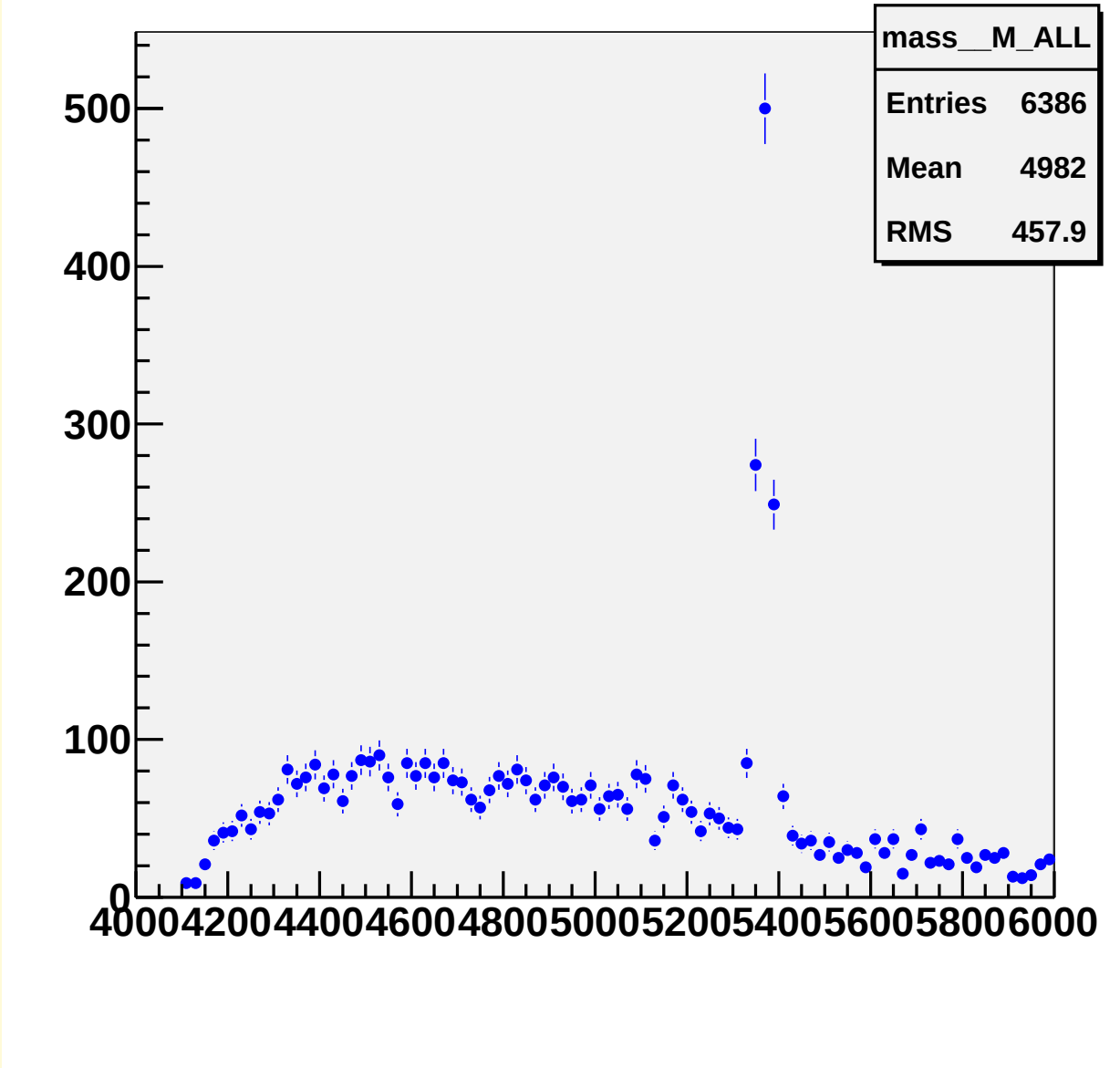
Plot Tool

`CombineParticles` and `FilterDesktop` have an integrated PlotTool.

```
from Configurables import LoKi__Hybrid__PlotTool as PlotTool
import GaudiKernel.SystemOfUnits as Units
bs2jpsiphi.HistoProduce = True
bs2jpsiphi.addTool( PlotTool("DaughtersPlots") )
bs2jpsiphi.DaughtersPlots.Histos = { "P/1000"  : ('momentum',0,100) ,
                                     "PT/1000" : ('pt_%1%',0,10,200) ,
                                     "M"       : ('mass in MeV_%1%_%2%_%3%
bs2jpsiphi.addTool( PlotTool("MotherPlots") )
bs2jpsiphi.MotherPlots.Histos = { "P/1000"  : ('momentum',0,100) ,
                                  "PT/1000" : ('pt_%1%',0,10,200) ,
                                  "M"       : ('mass_%1%_%2%_%3%',4*Units.G
```

The syntax is the same as for the cuts.

Plot Tool


mass__M_ALL
Entries 6386
Mean 4982
RMS 457.9
500
400
300
200
100
0
4000
4200
4400
4600
4800
5000
5200
5400
5600
5800
6000

Quick Debugging

When an algorithm does not do what I expect I usually do the following :

1. **Look at timing table**
2. **Look at `CombineParticles` statistics**
3. **Counters**
4. **Use the Debug Tool**
5. **Look at plots**
6. **If everything fails, set the algorithm to `OutputLevel = 2` or even `1`.**
   - **Then use `Lbglimpse` to find the messages in the code**

More resources

FAQ: There is a **DaVinci** FAQ at

`https://twiki.cern.ch/twiki/bin/view/LHCb/FAQ/DaVinciFAQ`

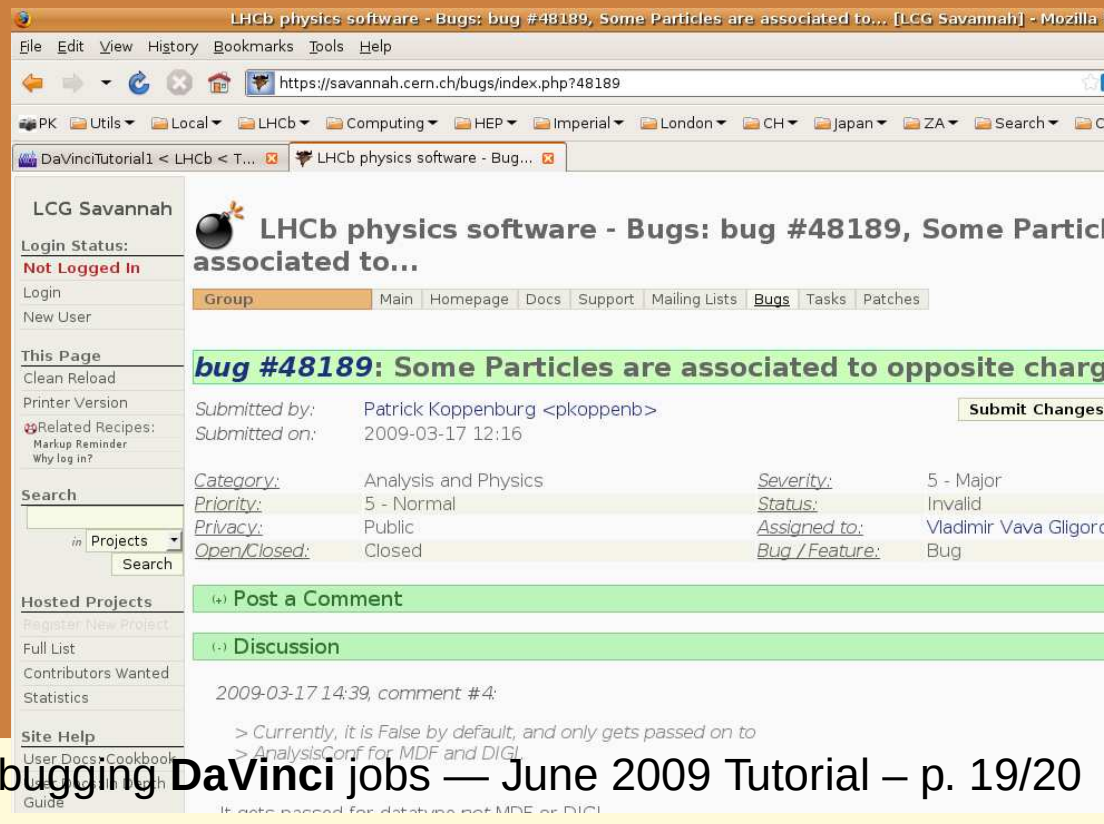
DaVinci ML: You can search in previous posts at

`https://groups.cern.ch/Pages/lhcb.aspx`

Savannah: Known bugs are reported at

`https://savannah.cern.ch/bugs/?group=lhcbcore`

- ➜ Check if your problem has been reported here
  - Once you're confident it's a bug, please report on the savannah tracker
  - This will help following up the bug



Conclusion

- The timing table tells you so much more than timing
- Look at the end of logfile statistics
- Switch on plotting
- Use debugging tools