
Summary of the athenaMP/Core/PMB Session

Paolo Calafiura

Multi-process Production

Douglas Smith
SLAC National Accelerator Lab.

Paolo Calafiura
Lawrence Berkeley Lab.

Production testing of Athena MP

- Athena MP has been in testing for awhile, but just as singular test jobs from patched releases.
- Stable releases to support AthenaMP have been around for a few months.
- Moving tests into production system, where stable tags are created, and tasks requested that turn into sets of jobs.
- At meeting in Napoli start of Feb., in discussion it was desired to see the ability of this to work for data production.
- A new task force has been created in the ADC group for Athena MP, and results presented earlier this week (and today).
- Goal is to show data production fully proven by the summer.

What is working...

- Production tests have been done with release 16.5.0.3
 - Multiprocess use of athena works, with caveats...
 - Reco jobs have been tested, from RAW to ESD and this works.
 - Except, because of a bug in the metadata, the validation step fails at the end, so you can only run with “omitvalid” flag on.
 - Also, there is bug in the production of ESD to AOD, and this fails.
 - So, normal production reco jobs actually don't work.
 - Also simu jobs don't work in 16.5.x.y. Releases.
- Confined to one specific type of job that works at this time.
- Bug fixes exist for these problems, and are going into new releases.

ConfigurableFactory update + a few related topics

Martin Woudstra (UMass)

A-team meeting

ATLAS Software & Computing Workshop

April 7, 2011

Reminder: the issue

The problem:

- _ Not all configuration dependencies are setup explicitly
 - Example: ToolA uses ToolB
 - the python that configures ToolA should make sure that ToolB is configured, e.g. by calling the corresponding python.
Often not done. Assumes someone else already did it
- _ Makes the configuration fragile

The solution:

- _ Every tool/service/algorithm configuration should **explicitly call** the configuration setup for all the tools/services it uses
- _ The ConfigurableFactory is a way to streamline and automate this

Abuse of MessageSvc and MsgStream Objects

Charles Leggett
Thomas Kittelmann

The Problem

- perfmon shows a lot of time spent inside MsgStream objects that are never printed out, because the job/algorithm logging level is set higher than the level of the MsgStream object. ie DEBUG messages:

```
...
MsgStream log(msgSvc,name());
log << MSG::DEBUG << "lets " << write << " a " << m_lot
    << " of " << m_stuff << " to " << debug << output << endreq;
```

- even worse:

```
...
MsgStream log(msgSvc,name());

while (counter < a_very_large_number) {
    important_var = do_some_useful_crap(m_crap);
    log << MSG::DEBUG << "the output of do_something with input "
        << m_crap << " is " << important_var << endreq;
}
```


How To Fix Your Code

- ~~it's easy!~~

```
if (m_log.level() <= MSG::DEBUG) {  
    m_log << MSG::DEBUG << "My output is now protected!"  
    << endreq;  
}
```

- if you want to be super anal:

```
#ifndef NDEBUG  
    if (m_log.level() <= MSG::DEBUG) {  
        m_log << MSG::DEBUG << "My output is now protected!"  
        << endreq;  
    }  
#endif
```

RAWtoESD: Top 10 Offenders



DEBUG

ToolSvc.TrigDataAccess	4606240
LArRawChannelBuilder	257518
ToolSvc.LArHVCablingTool	100459
MdtSubdetectorMap	88764
ToolSvc.LArRodDecoder	65058
InDetSCT_Clusterization	45890
TriggerTowerCollectionConverter	37130
StoreGateSvc	35107
ToolSvc.RpcPrepDataProviderTool	32836
MuonCalibAlg	32139

VERBOSE

ClassIDSvc	31110828
ToolSvc.RpcR0D_Decoder	1023886
ToolSvc.MuonTrackingGeometryBuilder	285477
ToolSvc.HLTCaloFEB	219080
ToolSvc.TileCondToolEmscale	158977
ToolSvc.MDTCablingDbTool	132200
LArRawChannelBuilder	128751
SCT_ClusterContainerCnv	91691
TRT_DriftCircleContainerCnv	82670
ToolSvc.MGM_AlignmentDbTool	38176

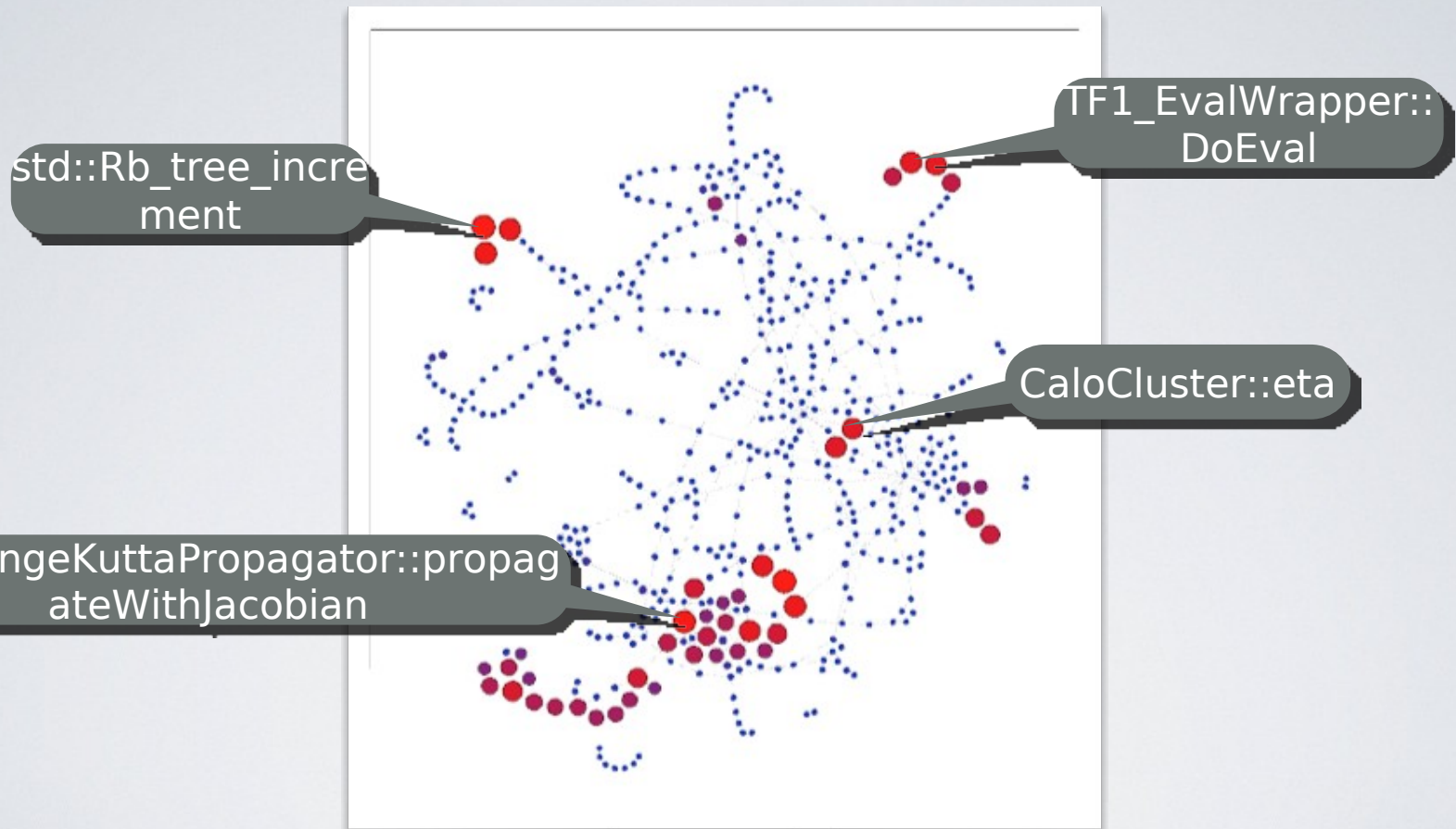
Callgraph analysis & Performance tools developments

Roberto A. Vitillo

Lawrence Berkeley National Laboratory

ATLAS Software and Computing Workshop, 7
April 2011

CallGraph analysis



Notice the chains: usually a function calls only one "heavy" function

Performance tools

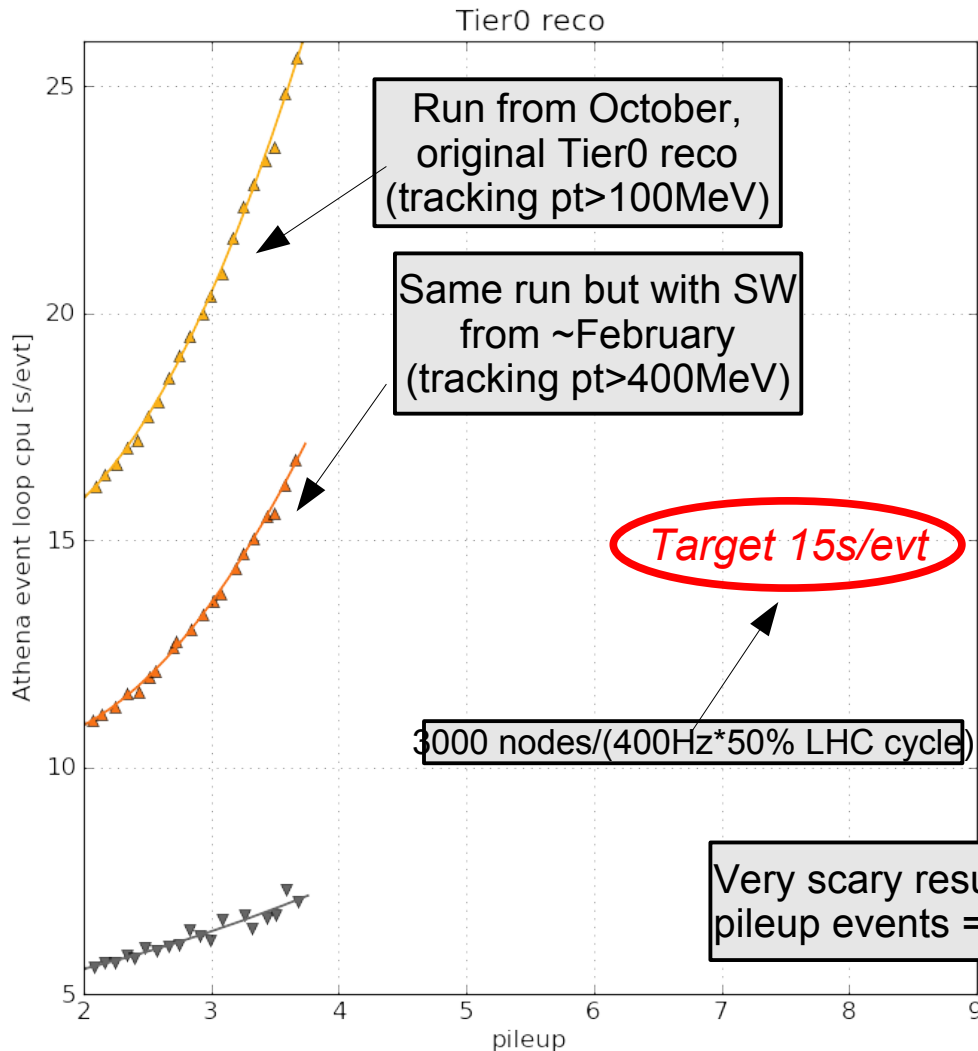
- Collaboration with Google: David Levinthal & Stephane Eranian
- Short term goal: use kcache-grind to visualize perf-events reports
 - Benefits: performance wise an order of magnitude faster than using instrumented code
- Long term goal: develop an open source visualizer that permits to interpret and to show collected data with emphasis on OO applications

PMB update: Brief updates on Tier0 performance and other developments

NB: For once we skip the usual round of domain reports

*Thomas Kittelmann
University of Pittsburgh
ATLAS S&C week
Core/PMB session¹⁴
April 7, 2011*

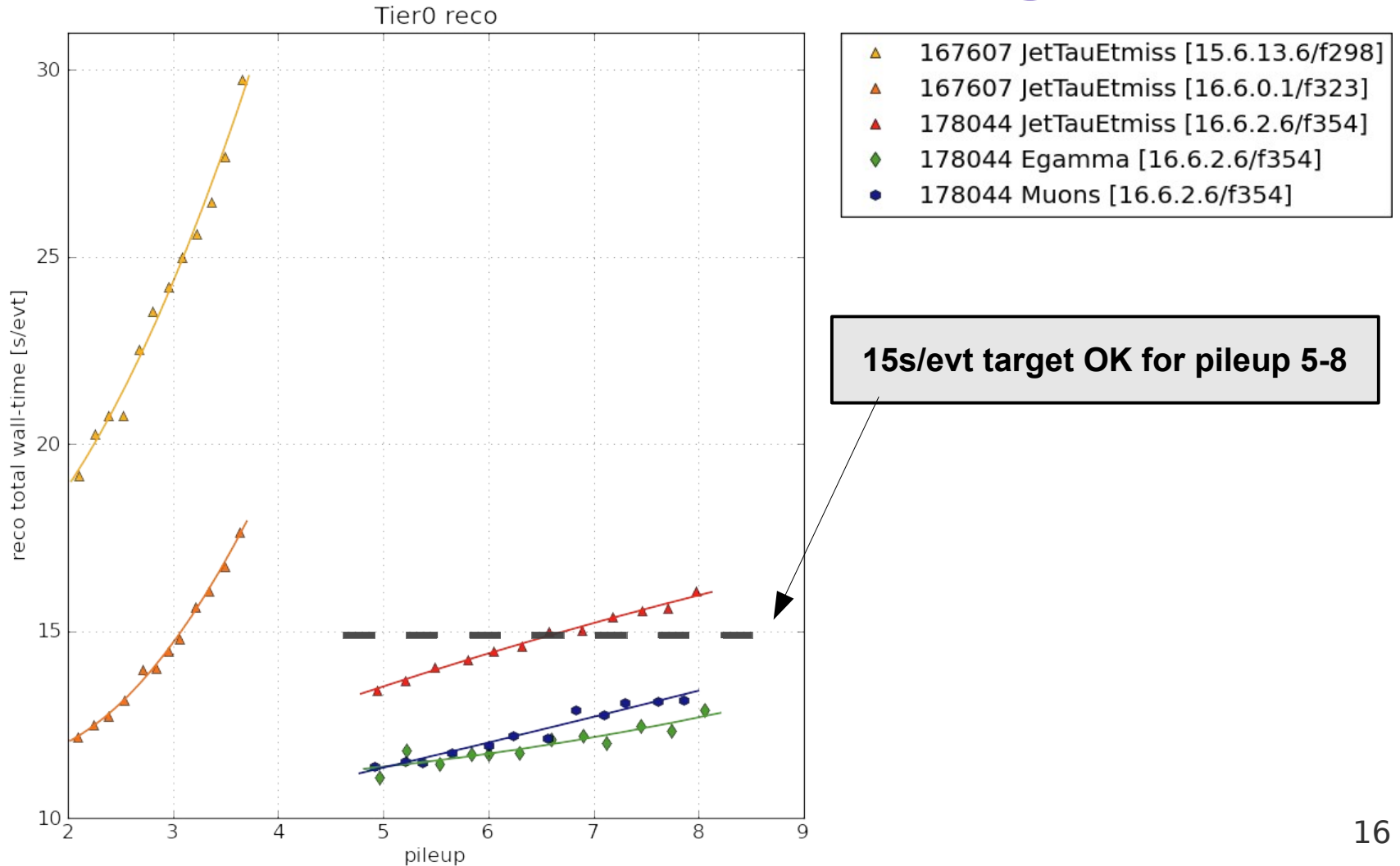
CPU@Tier0: 1-2 months ago...



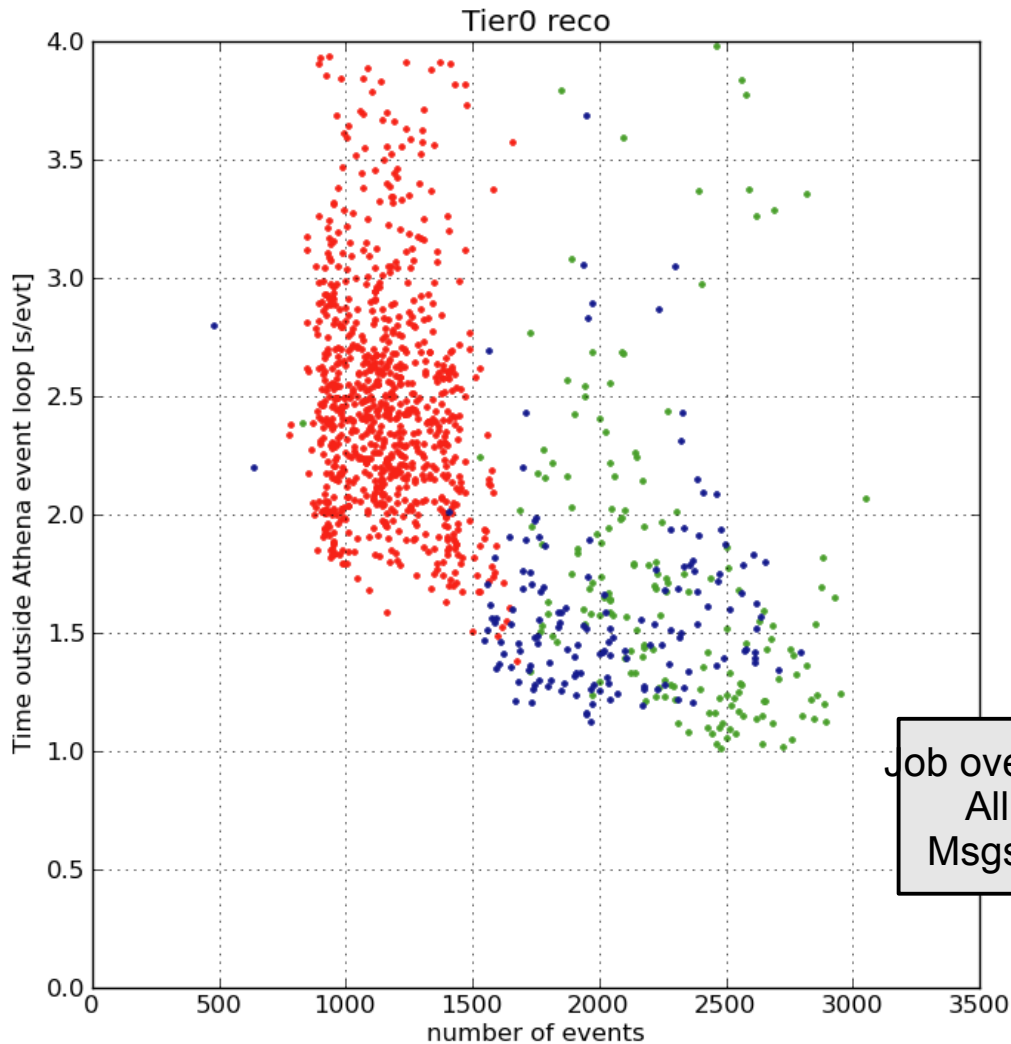
- ▲ 167607 JetTauEtmis [15.6.13.6/f298]
- ▲ 167607 JetTauEtmis [16.6.0.1/f323]
- ▼ 167607 ZeroBias [16.6.2.2/f341]



All in all: Waltime/evt@tier0



Time spent outside Athena event loops



- 178044 JetTauEtmis [16.6.2.6/f354]
- 178044 Egamma [16.6.2.6/f354]
- 178044 Muons [16.6.2.6/f354]

Job overhead still important at T0, but no disaster.
All improvements welcome, incoming are
Msgsvc abuse cleanup + GeoModel speedup

New “semi-detailed” mode for PerfMon

- A few highlights:
 - Enables per-component but not per-evt monitoring
 - Only top consumers in log-file, full output inside perfmon ntuple.
 - Light-weight enough to be enabled by default in reco jobs.
 - No double-counting of contributions (like when AlgA initialize triggers ToolB initialize).
 - Comes with python modules for easy parsing (including nitty gritty like versioning and backwards compat.)!
 - Several other new variables: Python cfg time, evt loop wall time, mean of VMEM/RSS throughout job, ...
- In dev nightlies + 16.6.3.5. Soon to be enabled by default.
- To be used by Ilija's scripts summarizing per-run info in AML.
- Refer to wiki for more information:

<https://twiki.cern.ch/twiki/bin/viewauth/Atlas/PerfMonSD>

The Economics of CPU Optimization

- It takes several thousand functions to account for 80% of reconstruction CPU usage (D. Levinthal)
- Developers time is more valuable than CPU time

yes, but

- ATLAS today uses 66K cores (M. Ernst) at a cost of 200-500 CHF/core/year (Y. Yao)

1% CPU gain > 100K CHF/year savings

Spending a week to gain 0.1% CPU
very profitable use of your time!