

Running jobs on the Farm(s)

Farms and batch, LSF, the STAR Scheduler and FileCatalog

STAR available farms

Currently, two main processing farms

- the RHIC Computing Facility (**RCF**) at BNL
RedHat 8.0
30 interactive nodes
154 dual Pentium III or IV (up to 2.4Ghz)
- the Parallel Distributed Systems Facility (**PDSF**) at NERSC
RedHat 7.2
7 interactive nodes
190 dual Pentium III nodes (650 Mhz to 1.8 GHZ)

Ratio interactive/non-interactive
RCF~20% PDSF~4%

Jerome LAURET, BNL

STAR Regional Meeting, Dubna (Russia) – Nov 2003

Batch Systems

Running interactively is NOT an option

- Don't know how many processes are already running, cannot assign priorities
- Jobs compete with each other (loss of performance due to swapping). No load balancing possible ...
- Legitimate interactive work jeopardized (not fair) (compilation/code development, testing, editor ...)

Alternative, using “a” batch system

- PDSF and RCF : LSF
versatile, flexible, ... but commercial
- Others: PBS, Condor, NQS, ...
all free, not as flexible, Condor a “plus” as really close

Before using LSF ...

The Golden rules to batch happiness

- **1 - ALWAYS try one job interactively first**

- **2 - DO NOT create a .tcshrc file**

tcsh reads .cshrc file if the other one is missing, batch may start using csh (not tcsh)

- **3 - Use explicit STAR_LEVELS**

A batch starts a new shell → The STAR environment is initialized to the site default (**pro** at the RCF, **new** at PDSF)

In your .cshrc

```
setenv STAR_LEVELS dev
```

Before executing your program using command line setup

```
starver SL03g
```

```
starpro
```

...

- **4 - Avoid using LSF short cut submission command**

~~% bsub -q 'some command'~~ (it is the #1 source for user confusion). Use an intermediate shell script (+ rule above)

So, how do I use it ?

Life is simple ...

http://www.star.bnl.gov/STAR/comp/train/using_lsf.html

Submit a script

```
% bsub -q queueName -o result.log -e result.err myScript.csh
```

Show what is running bjobs

Killing a job bkill JobID

Killing all jobs bkill 0

URL has more tips and tricks, as well as detailed how-to for writing a script ready for batch jobs ...

BEWARE : not all queues have the same priority and/or restrictions

Jerome LAURET, BNL

STAR Regional Meeting, Dubna (Russia) – Nov 2003

So, how do I use it ?

Life is not that simple ...

A batch script can be as simple as

```
#!/bin/csh
stardev                # golden rule No 3
cd myWorkDirectory/   # ...
root4star -q -b 'mymacro.C
    (10, "/star/data09/reco/ppMinBias/ReversedReversld/P02ge/2001/355/st
    _physics_2355006_raw_0041.MuDst.root", "myresult.root")'
```

using a specific file as input and output-ing `myresult.root` containing some histograms `mymacro.C` extracts.

Immediate problems

- Submitting by hand may be inefficient (resources depends on sites)
- There are ~ 3 Million files (as many batch script to write ?? which one are “goo” ??). They are located in many places (sometimes change location)
- Have to deal with Batch technology / solution

Life is definitely not simple ...

Resources

http://www.star.bnl.gov/STAR/comp/train/using_lsf.html
<http://www.star.bnl.gov/STAR/comp/train/tut/LSF-PDSF-for-You/>

When too many jobs try to read the same data at the same time (or lots of small chunks everywhere), the disk vault (PDSF) or NFS servers (RCF) spend all the time trying to figure out whom to serve. Global impact ...

LSF Resources

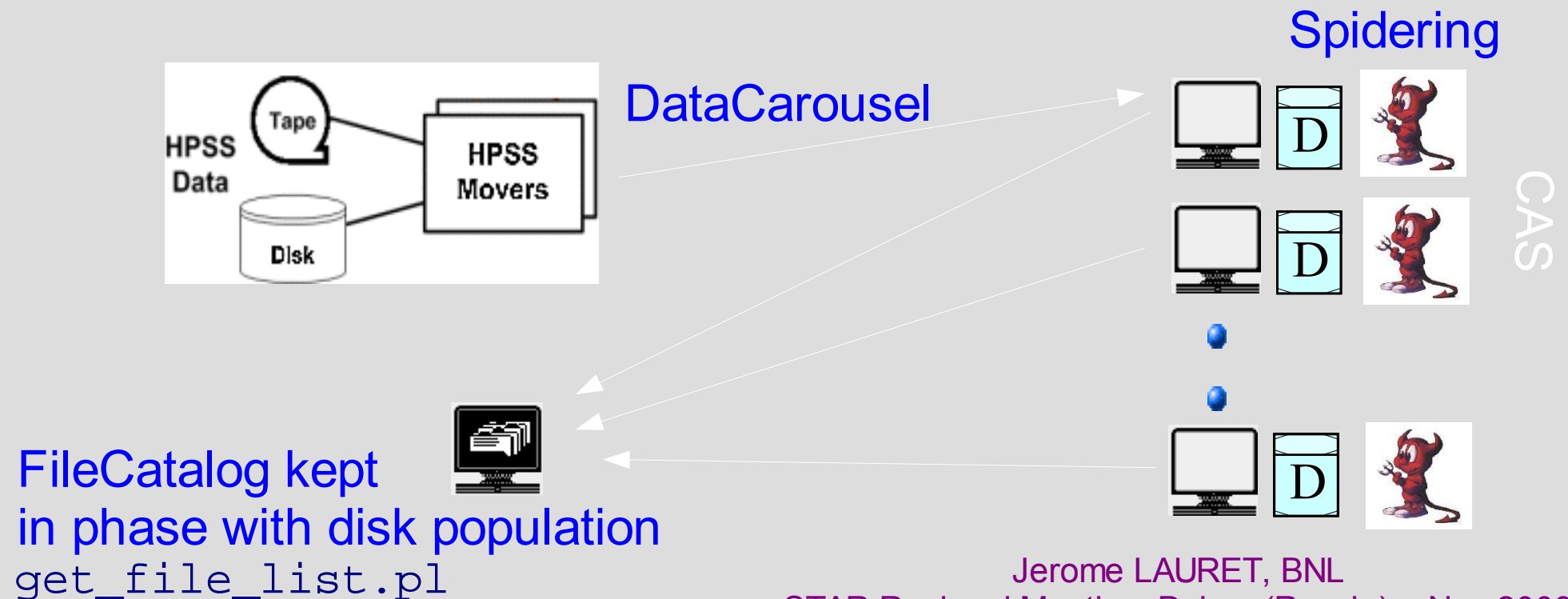
```
% bsub -q star_cas_short -o result.log -e result.err -R "rusage[sd1=50:sdu=10]" <yourJob>
```

```
% bsub -q medium -R "select[defined(dv27io)&&defined(dv34io)&&scratch>2000] rusage [dv27io=33:dv34io=33] " <your-Job>
```

Life is definitely not simple ...

File placement

- Files are inventoried into a FileCatalog and marked as bad as approved by PWG / PAC
- Files are located on 33 centralized disks at BNL, 100 distributed disks



Life is is good again ...

FileCatalog (replica catalog)

- STOP fighting with trying to locate files, STOP using `find`, recursive search, wildcarded `ls` etc ...
- Use `get_file_list.pl` (one API to the STAR FileCatalog)

```
% get_file_list.pl -keys path,filename -cond
  trgsetupname=UPCCombined
% get_file_list.pl -keys path,filename -cond
  trgsetupname=UPCCombined,storage=NFS, filetype=daq_reco_MuDst
% get_file_list.pl -keys storage -cond trgsetupname=UPCCombined
% get_file_list.pl -keys keyword {-distinct} {-alls} {-cond
  conditions}
```

<http://www.star.bnl.gov/STAR/comp/sofi/FileCatalog/>

<http://www.star.bnl.gov/STAR/comp/train/tut/> FileCatalog tutorial

Hypernews : catalog-hn@www.star.bnl.gov

Life is is good again ...

The STAR Scheduler

- It presents itself as an **abstract layer** on top of “a” batch system : **users writes** a job description of his “**intent**” in XML (eXtensible Markup Language)
- Fully **integrated with the STAR FileCatalog** has the ability to **use dd and/or centralized storage, no need to keep track of files**
- deals with several batch systems AND/OR site resources (**no need to know about implementation specifics ; you are ready for running on the Grid**)
- It is capable of doing resource brokering !!! (multiple choices get resolved according to policies)
- You have to comply with a few XML rules
- It DOES NOT start the coffee machine in the morning :-) ...

How does a XML looks like ??

SchedulerExample.xml

- XXX wants to execute a root macro on all the MuDST with minbias trigger and collision deuteron-Gold at 200 GeV, production P03ia, ReversedFullField
- The output of our macro will be a root file containing a histogram, we will move it to a specific location when done

```
<?xml version="1.0" encoding="utf-8" ?>
<job maxFilesPerProcess="25" >
  <command>stardev
    root4star -b -q /star/u/XXX/Analysis/dAuTest/runFromMuDst.C\
    (\ "$FILELIST" , \ "$SCRATCH/MyAnalysisdAu_MinBias.$JOBID.root" , 2E5) </command>

  <output fromScratch="*.root" toURL="file:/star/data02/pwg/XXX/work/dAu/" />

  <stdout URL="file:/star/u/XXX/Analysis/dAu/MyAnalysisdAu_MinBias.$JOBID.out" />
  <stderr URL="file:/star/u/XXX/Analysis/dAu/MyAnalysisdAu_MinBias.$JOBID.out" />

  <input
    URL="catalog:star.bnl.gov?production=P03ia,filetype=daq_reco_MuDst,storage!=hpss,trgsetupname=d
    AuMinBias,magscale=ReversedFullField" nFiles="all" />
</job>
```

Jerome LAURET, BNL

STAR Regional Meeting, Dubna (Russia) – Nov 2003

How does a XML looks like ??

SchedulerExample.xml

- XXX wants to execute a root macro on all the MuDST with minbias trigger and collision deuteron-Gold at 200 GeV, production P03ia, ReversedFullField
- The output of our macro will be a root file containing a histogram, we will move it to a specific location when done

<?xml version="1.0" encoding="utf-8" ?> ← Just write this ...

<job maxFilesPerProcess="25" >
 <command>stardev

root4star -b -q /star/u/XXX/Analysis/dAuTest/runFromMuDst.C\

→ (" \$FILELIST\"," \$SCRATCH/MyAnalysisdAu_MinBias.\$JOBID.root\","2E5)</command>

The macro takes a file list ... Check the MuDst macros/ directory

<output fromScratch="*.root" toURL="file:/star/data02/pwg/XXX/work/dAu/">

<stdout URL="file:/star/u/XXX/Analysis/dAu/MyAnalysisdAu_MinBias.\$JOBID.out"/> This is a
 <stderr URL="file:/star/u/XXX/Analysis/dAu/MyAnalysisdAu_MinBias.\$JOBID.out"/> Catalog query

<input
 URL="catalog:star.bnl.gov?production=P03ia,filetype=daq_reco_MuDst,storage!=hpss,trgsetupname=d
 AuMinBias,magscale=ReversedFullField" nFiles="all"/>

</job>

Jerome LAURET, BNL

STAR Regional Meeting, Dubna (Russia) – Nov 2003

Same (no Catalog)

```
<?xml version="1.0" encoding="utf-8" ?>
<job maxFilesPerProcess="25" >
  <command>root4star -b -q /star/u/XXX/Analysis/dAu/runFromMuDst.C\
    (\ "$FILELIST\" , \ "/star/data02/pwg/XXX/work/dAu/MyAnalysisdAu_MinBias.$JOBID.root\" , 2
    E5)</command>
```

This will cause direct IO (bad !!)

If data02 runs out of space, your job(s) will die miserably ...

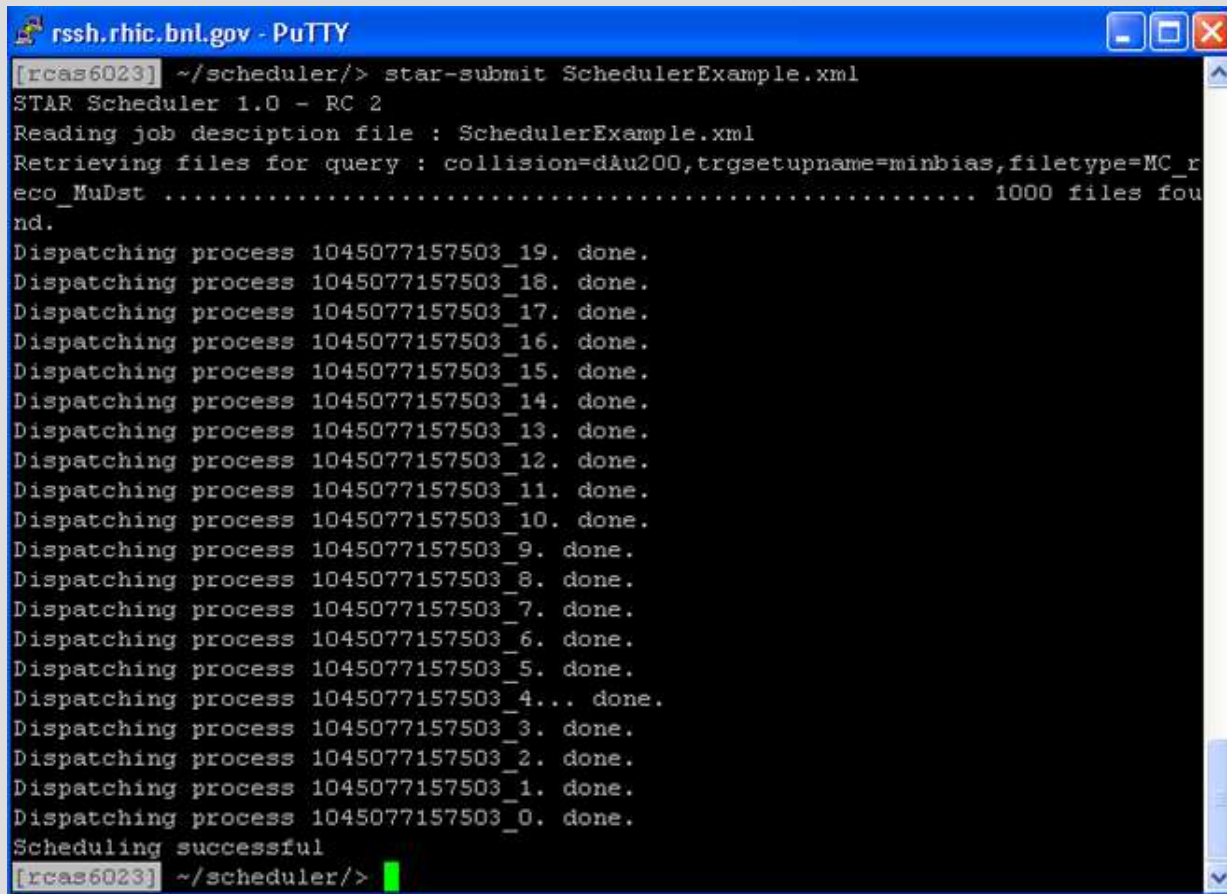
```
<stdout URL="file:/star/u/XXX/Analysis/dAu/MyAnalysisdAu_MinBias_Full.$JOBID.out"/>
<stderr URL="file:/star/u/XXX/Analysis/dAu/MyAnalysisdAu_MinBias_Full.$JOBID.out"/>
```

```
<input URL="file:/star/data16/reco/dAuMinBias/Revers*/P03ia/2003/025/*.MuDst.root" />
<input URL="file:/star/data16/reco/dAuMinBias/Revers*/P03ia/2003/026/*.MuDst.root" />
<input URL="file:/star/data16/reco/dAuMinBias/Revers*/P03ia/2003/027/*.MuDst.root" />
<input URL="file:/star/data16/reco/dAuMinBias/Revers*/P03ia/2003/028/*.MuDst.root" />
<input URL="file:/star/data16/reco/dAuMinBias/Revers*/P03ia/2003/033/*.MuDst.root" />
<input URL="file:/star/data16/reco/dAuMinBias/Revers*/P03ia/2003/034/*.MuDst.root" />
</job>
```

If the files are moved, or some are marked corrupted, you will learn about it the hard way ...

What to do next ...

```
% star-submit SchedulerExample.xml
```



```
rssh.rhic.bnl.gov - PuTTY
[rcas6023] ~/scheduler/> star-submit SchedulerExample.xml
STAR Scheduler 1.0 - RC 2
Reading job description file : SchedulerExample.xml
Retrieving files for query : collision=dAu200, trgsetupname=minbias, filetype=MC_r
eco_MuDst ..... 1000 files found.
Dispatching process 1045077157503_19. done.
Dispatching process 1045077157503_18. done.
Dispatching process 1045077157503_17. done.
Dispatching process 1045077157503_16. done.
Dispatching process 1045077157503_15. done.
Dispatching process 1045077157503_14. done.
Dispatching process 1045077157503_13. done.
Dispatching process 1045077157503_12. done.
Dispatching process 1045077157503_11. done.
Dispatching process 1045077157503_10. done.
Dispatching process 1045077157503_9. done.
Dispatching process 1045077157503_8. done.
Dispatching process 1045077157503_7. done.
Dispatching process 1045077157503_6. done.
Dispatching process 1045077157503_5. done.
Dispatching process 1045077157503_4... done.
Dispatching process 1045077157503_3. done.
Dispatching process 1045077157503_2. done.
Dispatching process 1045077157503_1. done.
Dispatching process 1045077157503_0. done.
Scheduling successful
[rcas6023] ~/scheduler/>
```

In the directory where you executed star-submit, a lot of .csh and .lis file will appear ...

They were ALL submitted to LSF

Tip1: use 'bjobs' to see them

Tip2: the first line of each script contains the submission cmd

Unless the jobs crashes, none of those temporary files are useful

Write them on a scratch space ...

Jerome LAURET, BNL

STAR Regional Meeting, Dubna (Russia) – Nov 2003

How did it do that ??

- It parsed your XML and memorized your intent
- It resolved your query or wildcard
- It divided the list of files in N jobs of each 25 files ...
Resource Brokering happened at this stage i.e. each .csh was produced to run optimally according to policies
- Not that since there was no specification (in the first example) of where the result should end up, it can in principle start ANYWHERE in the world ...

YES, the Scheduler IS Grid aware !!

Tips and notes

- Golden rule #2 applies (batch are csh scripts)
- Golden rule #3 applies (first example do use explicit `stardev` command)
- DO NOT set the LSF environment yourself. If you overwrite it, you are on your own ...
- When you use a Catalog query, try from the command line FIRST (`get_file_list.pl`)
- If you use the documented `preferStorage=NFS`, you have to specify `storage=NFS` in the Catalog query

Summary and more reading ...

- **Batch** is the way to go for large jobs. **Use it !!**
- The “**golden rules**” can save you days of struggle
- Submitting jobs properly can become an “expert” model
- In STAR, **the Scheduler** shields the user from the details (site specific, file placement)
- You are one step away from **running on the Grid ...**

`http://www.star.bnl.gov/STAR/comp/Grid/scheduler/
.../STAR/comp/Grid/scheduler/pres/SchedulerTutorial.ppt
.../STAR/comp/Grid/scheduler/faq.html
.../STAR/comp/meet/CM200308/STAR_Scheduler_A_tutorial.ppt`

Hypernews : `scheduler-hn@www.star.bnl.gov`