Group away day - pyCBC worksheet
Sebastian and Frank - 27/02/2015

So you want to run an offline matched filtered analysis for CBC sources on detector data!?

In this worksheet you will learn the following;

1. How to generate a grid proxy so you can login to ligo clusters
2. Login to geo using gsissh
    a. geo address - geo2.arcca.cf.ac.uk
3. Source the local LALsuite and pycbc
    a. source /scratch/LIGO/Apps/LALSuite/Master/etc/lalsuiterc
4. Create a directory to store your runs.
5. Create a workflow
6. Run pegasus planning script
7. Submit the pegasus job [well, you might not actually want to do this today]
8. Monitor jobs
9. Study Results Page [which magically, you will be able to do today]

1.a

Create a proxy

```
$ ligo-proxy-init albert.einstein
Your identity: albert.einstein@LIGO.ORG
Enter pass phrase for this identity:
Creating proxy ..................................... Done
Your proxy is valid until: Mar 1 11:39:45 2015 GMT
```

1.b

Now you have a proxy you can log in to any LIGO cluster.
Note you can also access clusters through **ssh** using the new command

```
$ ssh albert.einstein@ssh.ligo.org
```

which returns an interactive list of all the clusters you can login to.

1.c

Next login to geo2

```
$ gsissh geo2.arcca.cf.ac.uk
```

We have a cluster wide copy of LALsuite and pyCBC that we can use, thanks to Paul.
To set up your shell environment source Paul's script with

1.d

```
$ source /scratch/LIGO/Apps/LALSuite/Master/etc/lalsuiterc
Loading modules...Done.
Sourcing glue, lalapps, lalburst, lalframe, lalinference, lalinspiral, lalmetaio, lalpulsar,
lalsimulation, lalstochastic, lal, lalxml, pycbc, pylal, Done.
Configuring ligo_data_find...Done.
```

1.e

Check that you can run some LAL code. Run the following code and check that you get the same output.

```
$ ligo_data_find -o H -t H1_ER_C00_AGG -T --url-type file
file://localhost/scratch/LIGO/Data/H/H1_ER_C00_AGG/1102/H-H1_ER_C00_AGG-1102863360-256.gwf
```

Now that you are logged onto a cluster and have LALSuite and pyCBC running let's start to setup an analysis! This worksheet follows closely the following webpage:
https://ldas-jobs.ligo.caltech.edu/~cbc/docs/pycbc/workflow/pycbc_make_coinc_workflow.html

1.f

Make a directory where we can store the files of today's worksheet

```
$ cd ~/
$ mkdir away_day_pycbc
```

1.g

Next we need to get some example configuration (.ini) files

```
$ cp /scratch/LIGO/Admin/LALSuite/Build/pycbc/pycbc/workflow/ini_files/example_pycbc.ini .
$ cp /scratch/LIGO/Admin/LALSuite/Build/pycbc/pycbc/workflow/ini_files/example_pipedown.ini .
$ cp /scratch/LIGO/Admin/LALSuite/Build/pycbc/pycbc/workflow/ini_files/example_inj.ini .
```

We need to change the "example_pycbc.ini" file because there is an out of date option (see 1.h), but in addition to that, feel free to look around and play with changing options if you feel adventurous.

1.h

```
$ vi example_pycbc.ini
```

Then type "/ cluster-before-veto" to find this line in the file.
Delete this line, then save and quit the file.

Next we need to set up so shell variables

1.i

```
$ export LOCAL_CONFIG_FILES="example_pycbc.ini example_inj.ini example_pipedown.ini"
```

1.j

The example script is set up to analyse 24 hours of S6 data. (And who knows, there might be a "signal" in there waiting to be discovered.)

```
$ export GPS_START_TIME=967593543
$ export GPS_END_TIME=967679943
```

1.k

Log file directory

```
$ export LOGPATH=/scratch/${USER}/away_day_pycbc/log
$ export PIPEDOWNTMPSPACE=/scratch/${USER}/away_day_pycbc/
$ mkdir -p $LOGPATH
```

1.l

Output webpage directory

```
$ export HTMLDIR=/home/${USER}/public_html/LVC
$ mkdir -p $HTMLDIR
```

1.m

Now we have everything setup to construct the workflow.
Generate a workflow using:

```
$ pycbc_make_coinc_workflow –local-config-files ${LOCAL_CONFIG_FILES} \
                           --config-overrides workflow:start-time:${GPS_START_TIME} \
                                              workflow:end-time:${GPS_END_TIME} \
                                              workflow:workflow-html-basedir:${HTMLDIR} \
                                              workflow:pipedown-log-path:${LOGPATH} \
                                              workflow:pipedown-tmp-space:${PIPEDOWNTMPSPACE}
```

This will create a directory with the name of the start and end GPS time you specified earlier.

1.n

Now we want to run the pegasus planning script

```
$ cd ${GPS_START_TIME}-${GPS_END_TIME}
```

1.o

Run the pegasus planning script, and you should see output similar to the following:
Note: To actually start the analysis you would run the command 'pegasus-run'
However this analysis would take too long and so you can skip to the link at the bottom of the work sheet to view results!

```
$ pycbc_basic_pegasus_plan weekly_ahope.dax $LOGPATH
Generating concrete workflow
2015.02.25 14:59:12.741 GMT:

I have concretized your abstract workflow. The workflow has been entered
into the workflow database with a state of "planned". The next step is
to start or execute your workflow. The invocation required is

pegasus-run  /home/spx8sk/away_day_pycbc/log/spx8sk/pegasus/weekly_ahope/run0001
```

Then you can run the command pegasus-run and you should see something like the following

```
[spx8sk@geo2 967593543-967597143]$ pegasus-run
/home/spx8sk/pycbc_tests/feb_away_day_tests/test1/log/spx8sk/pegasus/weekly_ahope/run0001

------------------------------------------------------------------------
File for submitting this DAG to Condor           : weekly_ahope-0.dag.condor.sub
Log of DAGMan debugging messages                 : weekly_ahope-0.dag.dagman.out
Log of Condor library output                     : weekly_ahope-0.dag.lib.out
Log of Condor library error messages             : weekly_ahope-0.dag.lib.err
Log of the life of condor_dagman itself          : weekly_ahope-0.dag.dagman.log

Submitting job(s).
1 job(s) submitted to cluster 6292804.
------------------------------------------------------------------------

Your workflow has been started and is running in the base directory:

  /home/spx8sk/pycbc_tests/feb_away_day_tests/test1/log/spx8sk/pegasus/weekly_ahope/run0001

*** To monitor the workflow you can run ***

  pegasus-status -l
/home/spx8sk/pycbc_tests/feb_away_day_tests/test1/log/spx8sk/pegasus/weekly_ahope/run0001

*** To remove your workflow run ***

  pegasus-remove
/home/spx8sk/pycbc_tests/feb_away_day_tests/test1/log/spx8sk/pegasus/weekly_ahope/run0001
```

You can also use the following to monitor your jobs.

```
$ condor_q
```

keeping an eye on your jobs

```
$ pegasus-status -l /home/spx8sk/away_day_pycbc/log/spx8sk/pegasus/weekly_ahope/run0001
```

```
$ pegasus-analyzer -f /home/spx8skaway_day_pycbc/log/spx8sk/pegasus/weekly_ahope/run0001
```

Depending on the details of the analysis, for example the during of the length of detector time you wish to search over (which `${GPS_END_TIME}-${GPS_START_TIME}`), you will start to see ouput after some time.

For the 24 hour analysis, The whole job took about 4-5 hours to run.

After the main analysis has been done, the script also does some post-processing, and produces plots which eventually end up in the public_html directory. If for any reason the post-processing fails then you can manually run the process of building the web page using the following

```
$ pycbc_write_results_page --config-file
/home/spx8sk/pycbc_tests/feb_away_day_tests/967593543-967679943/wip.ini
```

To view a results web page change check out my example

https://geo2.arcca.cf.ac.uk/~spx8sk/ahope/967593543-967679943/