Alexander Morton Brunel University Thursday, 05 March 2015

How to create nTuples with the “Brunel nTupiliser” Documentation

Part 1a - Users using Crab 2:

Within the *test/* subdirectory there is a configuration file called *crab.cfg*. Within the file contains the various parameters required in order to run the nTupiliser code in *test/nTupliserData_cfg.py*. The dataset (listed under the parameter *datasetpath*), the local working directory (*ui_working_dir*), and the output location on the Brunel Tier 2 storage element (*user_remote_dir*) will have to be set by the user.

It goes without saying that if the user wishes to do something different to the norm in the nTupilisation process, they will need to modify *test/nTupliserData_cfg.py* before running Crab 2.

Once the crab configuration file is set up, the usual formulas before running CMS Software will need to be cast – ie. executing “*>>> cmsenv*”. The user will need to have a valid CMS VO certificate (this can be checked by excuting “*>>> voms-proxy-init -voms cms*”).

To submit the job to the grid, run Crab 2. A short overview is given below, but if you encounter any difficulty, the relevant documentation for using Crab 2 can be found on the CMS Twiki.

i. Execute “*>>> crab -create*” to create the job, check the availability of the dataset selected and to prepare all the jobs for submission according to the selected job splitting specified by the configuration file.
ii. Execute “*>>> crab -submit*” to submit the jobs.
iii. Execute “*>>> crab -status -c <dir-name>*” to check the progress of the submitted jobs.
iv. Once all the jobs have completed execute “*>>> crab –getoutput -c <dir-name>*” to retrieve the log files of the jobs (just the log files as the output files are copied over to the storage element associated to the T2 specified).

Continue to Part 2.

Part 1b - Users using Crab 3:

COMING SOON – once the config file works properly with it …

Part 2:

Once the output logs have been retrieved one needs to create a list of all the files produced from the Crab 2 jobs using the python script found in “*/skimMacros/*”. This is an important step as sometimes when jobs are re-submitted they produce extra files in the remote directory, so if one just runs over the entire contents of the directory, it can contain extra files and as such will double count. This script makes sure that one is only considering the correctly processed files. To run it the following needs to be entered into the console:

```
“>>> python listCorrectFilesInSRM.py <dir-Crab2-project> <fileListFileOutputName>”
```

Following this the following command must be run in order to put the file names into a format which root likes to handle (main cause I believe is that the original script was written for pion2 and wasn’t updated for pion – this corrects that):

```
>>> sed –i
's|srm://dc2-grid 64.brunel.ac.uk:8446/srm/managerv2?SFN=/dpm/|root://dc2-
grid-64.brunel.ac.uk//dpm/|g'
< fileListFileOutputName >
```

These file names can now be plugged into any root application on Pion and it should be able to access them just fine. The instructions on using the skimming program that copies the datasets from the T2 to the nfs is included below – as everything runs faster locally! After compiling the exe, source the setup file and run cmsenv (as it is implicit, I’ll state it explicitly, this program needs to be located in a CMSSW setup) - in that order! Afterwards, execute “*skims.exe*”:

```
>>> source setup.sh
>>> cmsenv
>>> ./skims.exe < fileListFileOutputName> <outputDataSetName>
```

N.B.

All “>>>” are referring to the start of a new line on the console – in case this wasn’t implicit enough.

Crab 2 and the skimming program have been run and tested only under CMSSW_5_3_20 by the author.

The author takes no responsibility for any unintended consequences stemming from use of these instructions. They are meant as a guide only, to provide a starting point and to give others documentation which he desperately lacked during his student days.