Alexander Morton Brunel University Wednesday, 16 March 2016

How to create nTuples with the “Brunel nTupliser” Documentation Run 2

Setup:

In the src/ directory of CMSSW, the Brunel nTupliser is downloaded from github through the following command:

“*>>> git clone –b <CMSSW_branch> git@github.com:davidcarbonis/NTupliser.git*”

Currently the branches available are “CMSSW_5_3_20” for Run 1 data/MC and “CMSSW_7_4_7”, “CMSSW_7_4_14” and “CMSSW_7_6_3” for Run 2 data. This document concerns using the nTupliser for Run 2 only.

For CMSSW_7_4_X branches, the following CMSSW dependencies must be downloaded:

Electron Identification MVA:

Download the package through the following command (in the CMSSW src directory):

“*>>> git git cms-merge-topic ikrav:egm_id_7.4.12_v1*”

Once all packages required have been downloaded, CMSSW will need to be compiled using the usual method:

“*>>> scram b”*

Part 1 – Creating nTuples using Crab3:

Within the *test/* subdirectory there are configuration files called Crab3Config_MC.py and Crab3Config_data.py which contain the necessary scripting to run the nTupilser code in *test/ nTupliserMC_cfg _cfg.py* or *test/ nTupliserData.py* respectively.

Before creating nTuples of either data or MC, it is recommended that the user check the following:

- The Global Tag, as this defines the offline conditions data. Whilst the cfg files for the nTupliser are usually kept up to date with the latest Global Tag, it is recommended that the user double check as their dataset may require differing conditions or the cfg file may be out of date. Conditions can be found here: https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideFrontierConditions
- The run range and luminosity mask for the Crab3 data config script. The various JSON files for luminosity and the relevant runs for each run of data taking can be found here: https://twiki.cern.ch/twiki/bin/viewauth/CMS/PdmV2015Analysis (N.B. Check elsewhere to confirm if there are any runs to avoid. At the time of writing, a number of runs for Run2015D have been excluded in the config file due to a poorly reconstructed beam spot).
- For MC datasets, check the the value of the bool flag "`isLHEflag`" in python/MakeTopologyNtuple_miniAOD_cfi.py. In the past, dedicated MC samples with vaired factorisation and renormalisation scales coherently vaired for the Matrix Element and Parton Shower steps of the generator were created. For Run 2 the per-event weights in the generator are available in miniAOD, allowing these systematics to be measured without the need for separate samples. The only exception to this are the tW samples which are created with Powerheg V1 (this process is not available in Powerheg V2 yet) as scale variations on the ME level are not possible via LHE weights in Powerheg V1. Thus, in order to avoid CMSSW from finishing prematurely with an error regarding a missing item, this flag for Powerheg V1 samples must be set to "`False`". All other cases, the flag can be set to "`True`"

The dataset (listed under the parameter `config.Data.inputDataset`), the local working directory (`config.General.requestName`), and the output dataset tag (`config.Data.outputDatasetTag`) will have to be set by the user.

A list of datasets available in miniAOD for the most recent (and thus relevant) campaigns, both MC and data, can be found on the CMS twiki (see below). Be sure to use a version of the nTupliser corresponding to the miniAOD campaign used to avoid incompatibilities. https://twiki.cern.ch/twiki/bin/view/CMSPublic/WorkBookMiniAOD

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 *nTupliserData_cfg.py*/ *nTupliserMC_cfg.py* before running Crab 3.

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 3. A short overview is given below, but if you encounter any difficulty, the relevant documentation for using Crab 3 can be found on the CMS Twiki.

i. Execute "*>>> crab submit -c <CRAB-config-file>*" to create and submit the jobs. The "*--dryrun*" argument will upload the job and run a local test, informing the user of whether it will run and the estimated completion times, before prompting the user so that the submitted job can proceed.
ii. Execute "*>>> crab status -d <dir-name>*" to check the progress of the submitted jobs. One can get a more detailed status report by added "—long" to the crab status command.
iii. Once all the jobs have completed execute "*>>> crab getoutput -d <dir-name>*" to retrieve the output of the jobs.

Once the output logs have been retrieved one needs to create a list of all the files produced from the Crab jobs. For Run 1 there was a script to produce a list of the T2 locations of the output produced using Crab 2. Given that Crab 3 is used now, and there is not an easy method to access the physical file names (PFNs) and their locations, currently one has to retrieve the files from the T2 prior to skimming. A solution to avoid having to copy large files over to the user's area and skim the files directly from their T2 storage location is being looked into.

In the directory "*/skimMacros/*", one finds the small program used to produce the skims used in the analysis code. 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*":

```
>>> make
>>> 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 3 and the skimming program have been run and tested only under CMSSW_7_4_7, CMSSW_7_4_14 and CMSSW_7_6_3 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.