User Manual

Utilization of PBS Job Scheduler and Singularity Container for Compute Intensive Tasks

prepared by

Server Coordination Team (SCT)

DEPARTMENT OF INFORMATION TECHNOLOGY

National Institute of Technology Karnataka, Surathkal

Version 1.0, March 2021

General Notes

<<Please read this page before proceeding to the section of your interest in this manual>>

(i) PBS is the only authorized medium of running any compute intensive tasks on the nodes mentioned in this document, any process started on the master node without the use of the PBS queue will be terminated.

(ii) Every user will have a default quota on the resources requested, the current quota limitations are as follows:

  a. The maximum number of resources which can be requested through a PBS job are **1 GPU and 6 CPUs.**
  b. The default **disk storage quota limits** are **50GB(+50GB)** for a **general user (UG/PG student)**, **100GB(+50GB)** for **other users (long term research users including UG/PG students, Ph.D. research scholars and faculty)**.
  c. The +50GB limit is a grace limit for additional storage, which will be available for a period of 7 days without any warning notices. In case the user exceeds the combined limit (including the additional grace storage limit) or does not reduce storage space to the allocated limit, then the user will not be able to write new files (The existing files can still be accessed but cannot be written to).
  d. If you wish to increase/modify the quota limitations for a specific purpose, please approach the System Administrator or Server Coordinator.

(iii) This manual contains the minimum configuration needed to run PBS jobs and is written with a focus on python-based applications. For more information about the PBS commands and configurations please refer the following links:

  a. https://www.nas.nasa.gov/hecc/support/kb/running-jobs-with-pbs-121/
  b. https://supercomputing.iitd.ac.in/?pbs

(iv) Only 1 job from a user can be active at a time, the other jobs submitted by the user will be queued up in the PBS job queue (queue name: **'itd'**). Only **8 concurrent jobs with 1 GPU each** can be run by **8 different users**.

(v) At present, resources (memory,cpus and gpus) can be requested from a single node at a time, they cannot be distributed. For example, ncpus=6:ngpus=1:host=dgx is possible but ncpus=4:host=master+1:ncpus=3:ngpus=1:host=dgx will not be effective.

(vi) A lot of commonly used libraries such as pip modules for deep learning frameworks such as tensorflow and pytorch along with openmpi for HPC applications have been installed, but the user wishes to install specific software please contact the System Administrator.

(vii) The alternate way to install software, create reproducible and portable environments is by using containers. Singularity is installed and is available with a lot of prebuilt images which also include docker images. These images can be used to create your own custom singularity container, install any required software, and run jobs through PBS without requiring the System Administrator's intervention in installing your required software.

(viii) An example of how to build a docker image using singularity and installing a python package is demonstrated in this manual. For additional documentation on Singularity, please refer the following links:

  a. https://singularity.lbl.gov/
  b. https://supercomputing.iitd.ac.in/?pbs

Table of Contents

1. Overview of Computing Resources which can be accessed through PBS

1.1. PBS Nodes

PBS (Portable Batch System) allows authorized users to schedule and run a job on the three nodes/servers in the Dept of Information Technology, NITK (National Institute of Technology Karnataka) Surathkal, whose configurations are in Table 1.

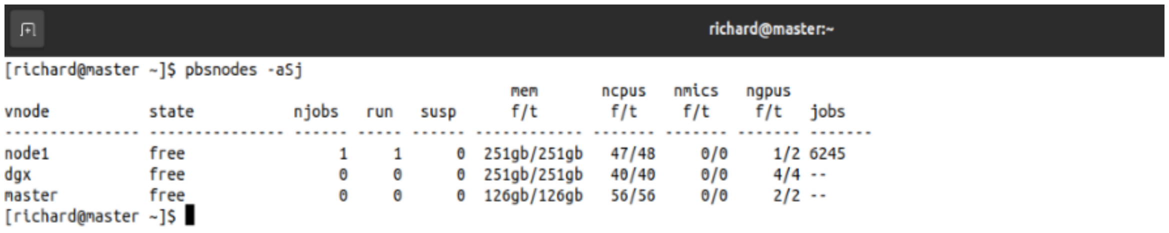
Table 1. Configuration of Nodes/Servers accessible through PBS

Node/Server Name	**IP Address**	**Number of CPUs**	**Number of GPUs (Type of GPUs)**	**Memory (in GB)**
master	10.100.52.148	56	2 (M40)	126
dgx	10.100.52.96	40	4 (V100)	251
node1	10.100.52.105	48	2 (V100)	251

The PBS scheduler runs from the master node; hence the users can submit and manage their jobs after logging into the master node. The following command is to view the status of resources available through PBS.

Command: pbsnodes –aSj

Output:



```
[richard@master ~]$ pbsnodes -aSj
                                                        mem       ncpus   nmics   ngpus
vnode           state           njobs   run   susp      f/t        f/t     f/t     f/t   jobs
--------------- --------------- ------ ----- ------ ------------ ------- ------- ------- -------
node1           free                 1     1      0  251gb/251gb   47/48     0/0     1/2 6245
dgx             free                 0     0      0  251gb/251gb   40/40     0/0     4/4 --
master          free                 0     0      0  126gb/126gb   56/56     0/0     2/2 --
[richard@master ~]$ 
```

1.2. Storage locations available to the users

Table 2. gives a list of storage locations available to the user and their purposes. These locations can be accessed from any on the three nodes. The backup directories are meant to store large datasets or project relevant deliverables. Please inform the system administrator before creating any such directories under /home2 and /home3.

Table 2. Storage locations for the users

Location	**Usage**
/home/<users_dir_name>	This the default home directory of a user (on the master node).
/home2/<backup_dir_name_for_storage>	The user can request for additional data storage at this location (on the dgx node)
/home3/<backup_dir_name_for_storage>	The user can request for additional data storage at this location (on the master node)

2. Building and Configuring a Singularity Container image

Ideally the process to create and use a Singularity Container has the following 3 steps:

i. Build (create) the container using a .sif (Singularity Image File) or a docker image for a plain OS like ubuntu with the bare minimum software installed or alternatively use a prebuilt docker image for a specific application like the docker image for tensorflow.
ii. Use the Singularity shell to execute commands or install software packages just like you would in your operating system's terminal.
iii. Execute your application (like a python program) in the container environment.

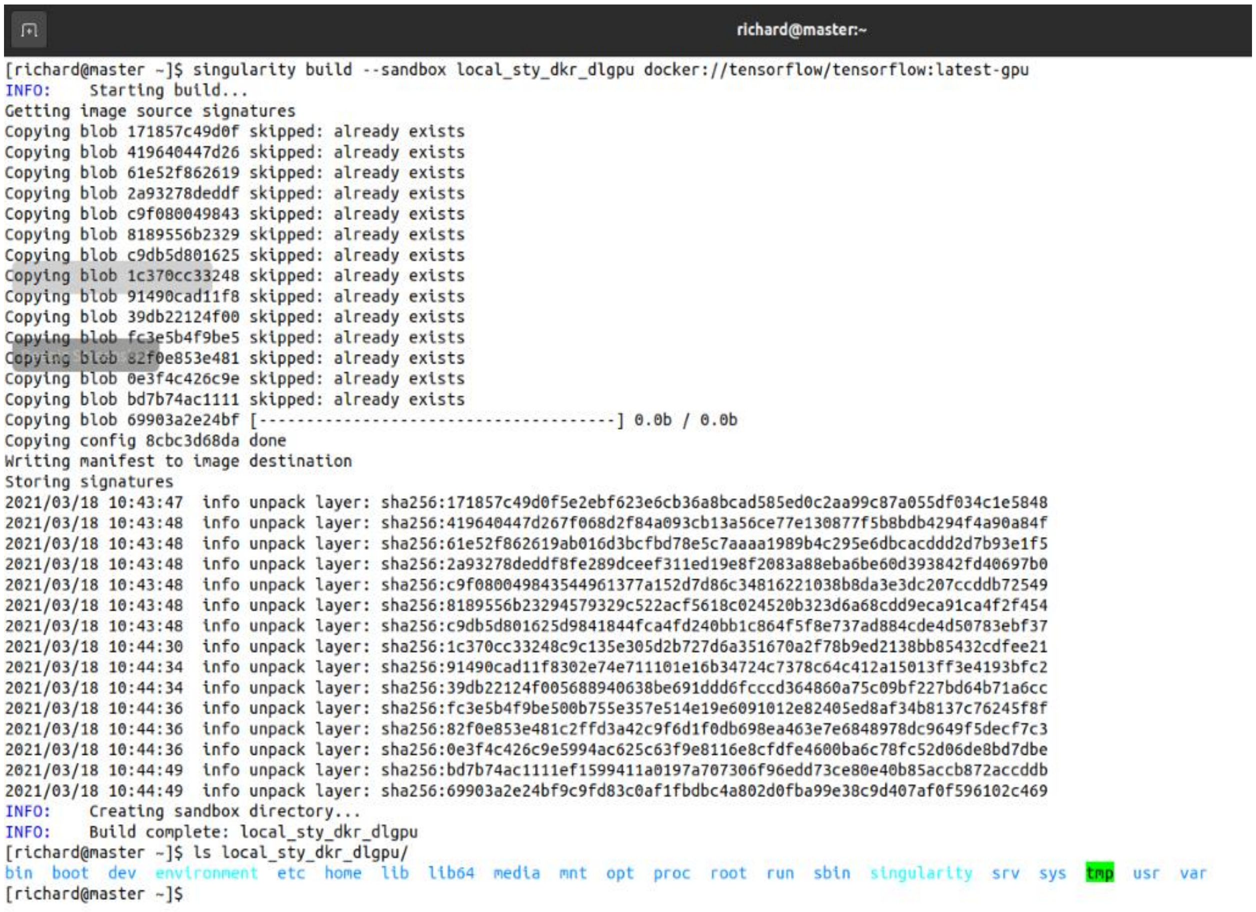
The following is a practical example of how you can build a Singularity Container using a docker image for tensorflow gpu, install additional pip (python) packages and execute a python application in the container.

2.1 Building the Singularity Container

Command: **singularity build** --sandbox <directory_name_of_your_container> <image_uri>

Example: **singularity build --sandbox local_sty_dkr_dlgpu docker://tensorflow/tensorflow:latest-gpu**

Output:



```
[richard@master ~]$ singularity build --sandbox local_sty_dkr_dlgpu docker://tensorflow/tensorflow:latest-gpu
INFO:    Starting build...
Getting image source signatures
Copying blob 171857c49d0f skipped: already exists
Copying blob 419640447d26 skipped: already exists
Copying blob 61e52f862619 skipped: already exists
Copying blob 2a93278deddf skipped: already exists
Copying blob c9f080049843 skipped: already exists
Copying blob 8189556b2329 skipped: already exists
Copying blob c9db5d801625 skipped: already exists
Copying blob 1c370cc33248 skipped: already exists
Copying blob 91490cad11f8 skipped: already exists
Copying blob 39db22124f00 skipped: already exists
Copying blob fc3e5b4f9be5 skipped: already exists
Copying blob 82f0e853e481 skipped: already exists
Copying blob 0e3f4c426c9e skipped: already exists
Copying blob bd7b74ac1111 skipped: already exists
Copying blob 69903a2e24bf [--------------------------------------] 0.0b / 0.0b
Copying config 8cbc3d68da done
Writing manifest to image destination
Storing signatures
2021/03/18 10:43:47  info unpack layer: sha256:171857c49d0f5e2ebf623e6cb36a8bcad585ed0c2aa99c87a055df034c1e5848
2021/03/18 10:43:48  info unpack layer: sha256:419640447d267f068d2f84a093cb13a56ce77e130877f5b8bdb4294f4a90a84f
2021/03/18 10:43:48  info unpack layer: sha256:61e52f862619ab016d3bcfbd78e5c7aaaa1989b4c295e6dbcacddd2d7b93e1f5
2021/03/18 10:43:48  info unpack layer: sha256:2a93278deddf8fe289dceef311ed19e8f2083a88eba6be60d393842fd40697b0
2021/03/18 10:43:48  info unpack layer: sha256:c9f080049843544961377a152d7d86c34816221038b8da3e3dc207ccddb72549
2021/03/18 10:43:48  info unpack layer: sha256:8189556b23294579329c522acf5618c024520b323d6a68cdd9eca91ca4f2f454
2021/03/18 10:43:48  info unpack layer: sha256:c9db5d801625d9841844fca4fd240bb1c864f5f8e737ad884cde4d50783ebf37
2021/03/18 10:44:30  info unpack layer: sha256:1c370cc33248c9c135e305d2b727d6a351670a2f78b9ed2138bb85432cdfee21
2021/03/18 10:44:34  info unpack layer: sha256:91490cad11f8302e74e711101e16b34724c7378c64c412a15013ff3e4193bfc2
2021/03/18 10:44:34  info unpack layer: sha256:39db22124f005688940638be691ddd6fcccd364860a75c09bf227bd64b71a6cc
2021/03/18 10:44:36  info unpack layer: sha256:fc3e5b4f9be500b755e357e514e19e6091012e82405ed8af34b8137c76245f8f
2021/03/18 10:44:36  info unpack layer: sha256:82f0e853e481c2ffd3a42c9f6d1f0db698ea463e7e6848978dc9649f5decf7c3
2021/03/18 10:44:36  info unpack layer: sha256:0e3f4c426c9e5994ac625c63f9e8116e8cfdfe4600ba6c78fc52d06de8bd7dbe
2021/03/18 10:44:49  info unpack layer: sha256:bd7b74ac1111ef1599411a0197a707306f96edd73ce80e40b85accb872accddb
2021/03/18 10:44:49  info unpack layer: sha256:69903a2e24bf9c9fd83c0af1fbdbc4a802d0fba99e38c9d407af0f596102c469
INFO:    Creating sandbox directory...
INFO:    Build complete: local_sty_dkr_dlgpu
[richard@master ~]$ ls local_sty_dkr_dlgpu/
bin  boot  dev  environment  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  singularity  srv  sys  tmp  usr  var
[richard@master ~]$
```

Note: sandbox option is not mandatory (but advisable). The Singularity container is now installed as the directory **local_sty_dkr_dlgpu**

2.2 Using the Singularity Container shell to install software and execute commands

Command: **singularity shell --writable <directory_name_of_your_container>**

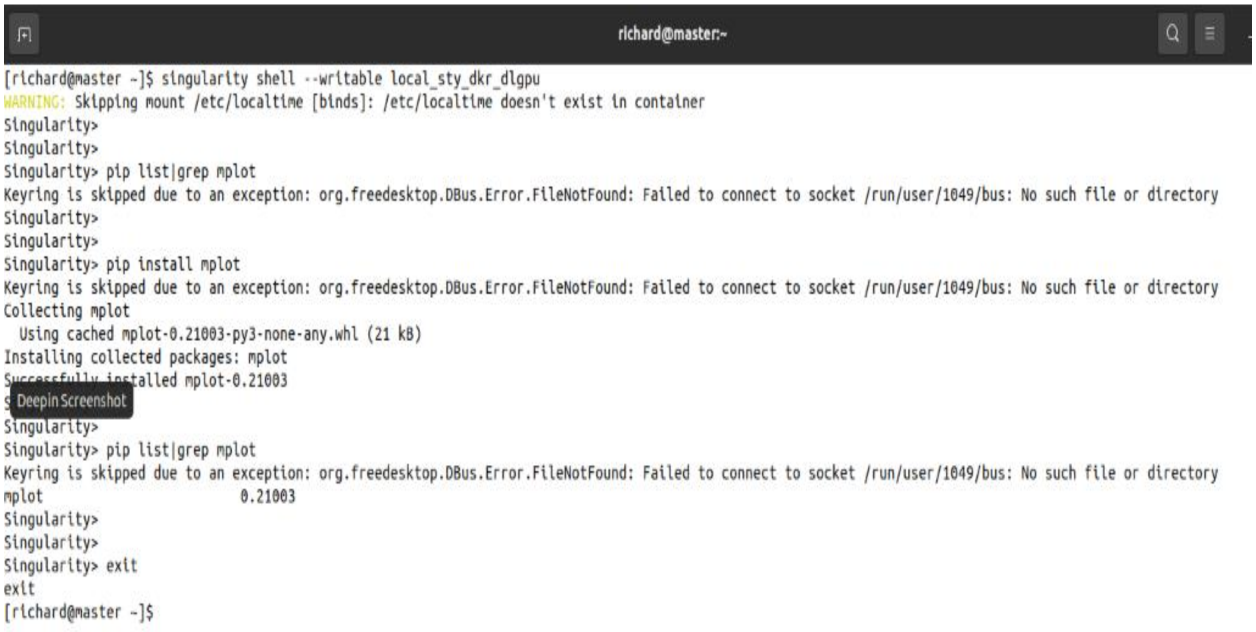
Example: **singularity shell --writable local_sty_dkr_dlgpu**

Singularity-prompt> pip list| grep mplot

Singularity-prompt> pip install mplot

Singularity-prompt>exit

Output:



```
[richard@master ~]$ singularity shell --writable local_sty_dkr_dlgpu
WARNING: Skipping mount /etc/localtime [binds]: /etc/localtime doesn't exist in container
Singularity>
Singularity>
Singularity> pip list|grep mplot
Keyring is skipped due to an exception: org.freedesktop.DBus.Error.FileNotFound: Failed to connect to socket /run/user/1049/bus: No such file or directory
Singularity>
Singularity>
Singularity> pip install mplot
Keyring is skipped due to an exception: org.freedesktop.DBus.Error.FileNotFound: Failed to connect to socket /run/user/1049/bus: No such file or directory
Collecting mplot
  Using cached mplot-0.21003-py3-none-any.whl (21 kB)
Installing collected packages: mplot
Successfully installed mplot-0.21003
S[illegible]
Singularity>
Singularity> pip list|grep mplot
Keyring is skipped due to an exception: org.freedesktop.DBus.Error.FileNotFound: Failed to connect to socket /run/user/1049/bus: No such file or directory
mplot                          0.21003
Singularity>
Singularity>
Singularity> exit
exit
[richard@master ~]$
```

Note:

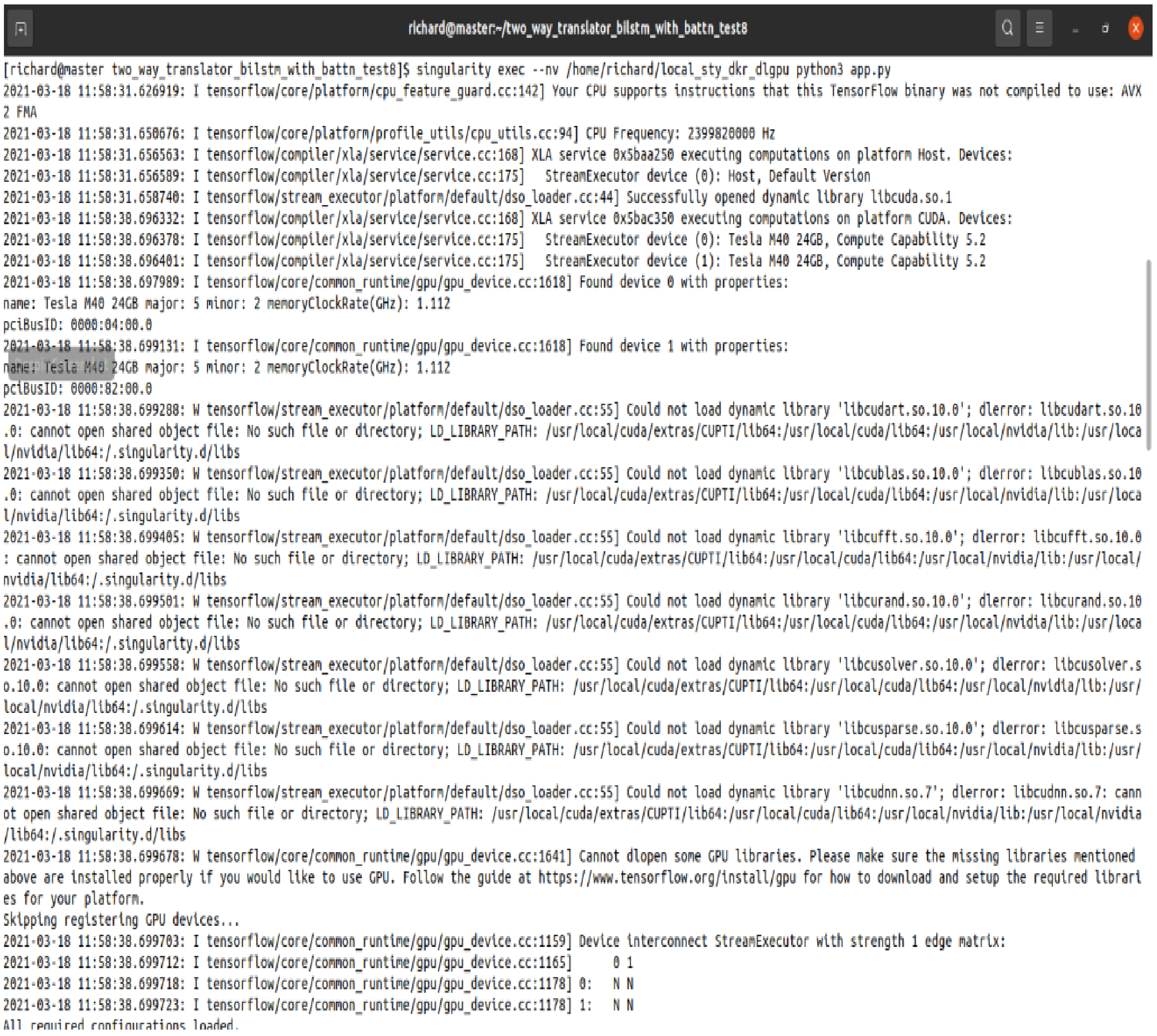
1. In the above example, the python package mplot is installed in the singularity container using pip installer. Other OS commands work in the Singularity prompt just as they would in the OS terminal.
2. The --writable option is to be used only for the first time to make the container directory writable.

2.3 Executing a user application or command using the container from outside the Singularity shell/prompt

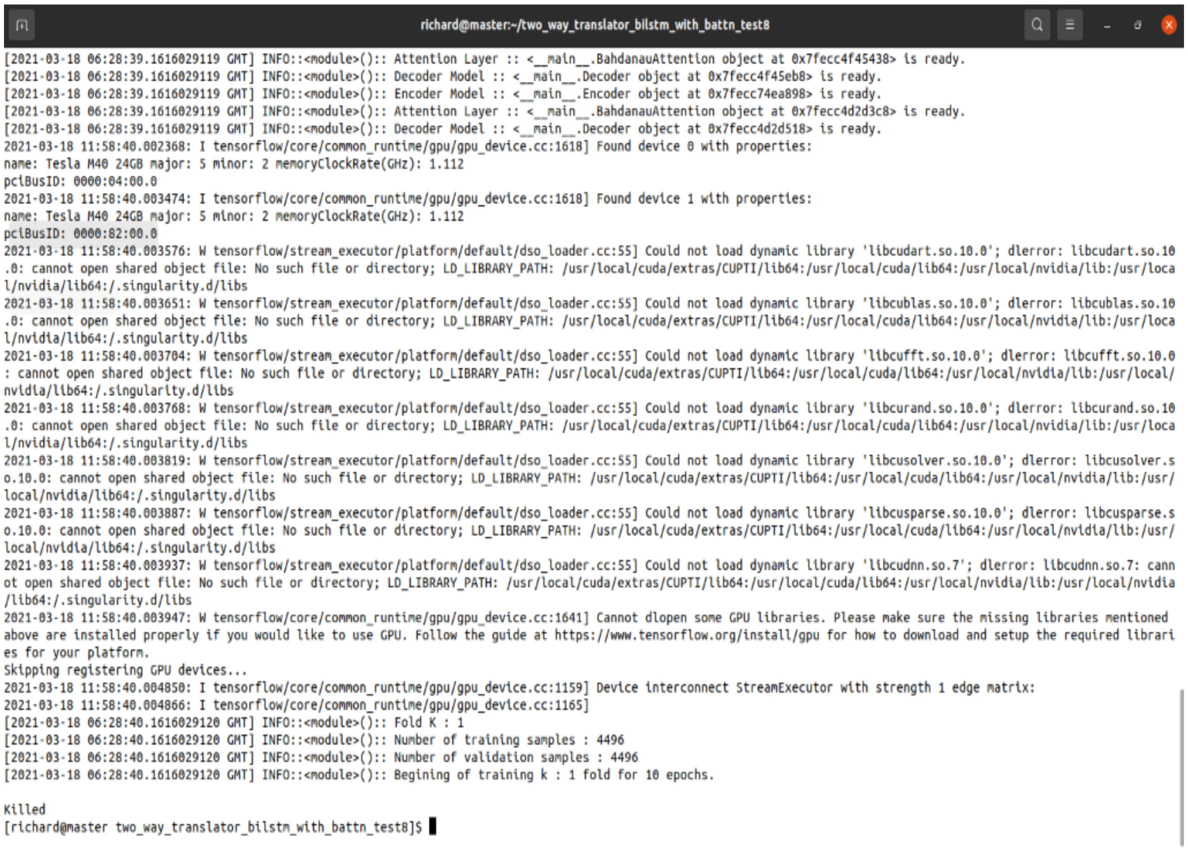
Command: **singularity** exec <directory_name_of_your_container> <command> [<file(s)/args>]

Example: **singularity exec** --nv /home/richard/local_sty_dkr_dlgpu python3 app.py

Output:



```
[richard@master two_way_translator_bilstm_with_battn_test8]$ singularity exec --nv /home/richard/local_sty_dkr_dlgpu python3 app.py
2021-03-18 11:58:31.626919: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX
2 FMA
2021-03-18 11:58:31.650676: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 2399820000 Hz
2021-03-18 11:58:31.656563: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x5baa250 executing computations on platform Host. Devices:
2021-03-18 11:58:31.656589: I tensorflow/compiler/xla/service/service.cc:175]   StreamExecutor device (0): Host, Default Version
2021-03-18 11:58:31.658740: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcuda.so.1
2021-03-18 11:58:38.696332: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x5bac350 executing computations on platform CUDA. Devices:
2021-03-18 11:58:38.696378: I tensorflow/compiler/xla/service/service.cc:175]   StreamExecutor device (0): Tesla M40 24GB, Compute Capability 5.2
2021-03-18 11:58:38.696401: I tensorflow/compiler/xla/service/service.cc:175]   StreamExecutor device (1): Tesla M40 24GB, Compute Capability 5.2
2021-03-18 11:58:38.697989: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: Tesla M40 24GB major: 5 minor: 2 memoryClockRate(GHz): 1.112
pciBusID: 0000:04:00.0
2021-03-18 11:58:38.699131: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 1 with properties:
name: Tesla M40 24GB major: 5 minor: 2 memoryClockRate(GHz): 1.112
pciBusID: 0000:82:00.0
2021-03-18 11:58:38.699288: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcudart.so.10.0'; dlerror: libcudart.so.10
.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/loca
l/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:38.699350: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcublas.so.10.0'; dlerror: libcublas.so.10
.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/loca
l/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:38.699405: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcufft.so.10.0'; dlerror: libcufft.so.10.0
: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/local/
nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:38.699501: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcurand.so.10.0'; dlerror: libcurand.so.10
.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/loca
l/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:38.699558: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcusolver.so.10.0'; dlerror: libcusolver.s
o.10.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/
local/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:38.699614: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcusparse.so.10.0'; dlerror: libcusparse.s
o.10.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/
local/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:38.699669: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcudnn.so.7'; dlerror: libcudnn.so.7: cann
ot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/local/nvidia
/lib64:/.singularity.d/libs
2021-03-18 11:58:38.699678: W tensorflow/core/common_runtime/gpu/gpu_device.cc:1641] Cannot dlopen some GPU libraries. Please make sure the missing libraries mentioned
above are installed properly if you would like to use GPU. Follow the guide at https://www.tensorflow.org/install/gpu for how to download and setup the required librari
es for your platform.
Skipping registering GPU devices...
2021-03-18 11:58:38.699703: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1159] Device interconnect StreamExecutor with strength 1 edge matrix:
2021-03-18 11:58:38.699712: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1165]      0 1
2021-03-18 11:58:38.699718: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1178] 0:   N N
2021-03-18 11:58:38.699723: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1178] 1:   N N
All required configurations loaded.
```



```
richard@master:~/two_way_translator_bilstm_with_battn_test8
[2021-03-18 06:28:39.1616029119 GMT] INFO::<module>():: Attention Layer :: <__main__.BahdanauAttention object at 0x7fecc4f45438> is ready.
[2021-03-18 06:28:39.1616029119 GMT] INFO::<module>():: Decoder Model :: <__main__.Decoder object at 0x7fecc4f45eb8> is ready.
[2021-03-18 06:28:39.1616029119 GMT] INFO::<module>():: Encoder Model :: <__main__.Encoder object at 0x7fecc74ea898> is ready.
[2021-03-18 06:28:39.1616029119 GMT] INFO::<module>():: Attention Layer :: <__main__.BahdanauAttention object at 0x7fecc4d2d3c8> is ready.
[2021-03-18 06:28:39.1616029119 GMT] INFO::<module>():: Decoder Model :: <__main__.Decoder object at 0x7fecc4d2d518> is ready.
2021-03-18 11:58:40.002368: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: Tesla M40 24GB major: 5 minor: 2 memoryClockRate(GHz): 1.112
pciBusID: 0000:04:00.0
2021-03-18 11:58:40.003474: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 1 with properties:
name: Tesla M40 24GB major: 5 minor: 2 memoryClockRate(GHz): 1.112
pciBusID: 0000:82:00.0
2021-03-18 11:58:40.003576: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcudart.so.10.0'; dlerror: libcudart.so.10
.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/loca
l/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:40.003651: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcublas.so.10.0'; dlerror: libcublas.so.10
.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/loca
l/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:40.003704: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcufft.so.10.0'; dlerror: libcufft.so.10.0
: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/local/
nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:40.003768: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcurand.so.10.0'; dlerror: libcurand.so.10
.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/loca
l/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:40.003819: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcusolver.so.10.0'; dlerror: libcusolver.s
o.10.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/
local/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:40.003887: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcusparse.so.10.0'; dlerror: libcusparse.s
o.10.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/
local/nvidia/lib64:/.singularity.d/libs
2021-03-18 11:58:40.003937: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcudnn.so.7'; dlerror: libcudnn.so.7: cann
ot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/local/nvidia
/lib64:/.singularity.d/libs
2021-03-18 11:58:40.003947: W tensorflow/core/common_runtime/gpu/gpu_device.cc:1641] Cannot dlopen some GPU libraries. Please make sure the missing libraries mentioned
above are installed properly if you would like to use GPU. Follow the guide at https://www.tensorflow.org/install/gpu for how to download and setup the required librari
es for your platform.
Skipping registering GPU devices...
2021-03-18 11:58:40.004850: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1159] Device interconnect StreamExecutor with strength 1 edge matrix:
2021-03-18 11:58:40.004866: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1165]
[2021-03-18 06:28:40.1616029120 GMT] INFO::<module>():: Fold K : 1
[2021-03-18 06:28:40.1616029120 GMT] INFO::<module>():: Number of training samples : 4496
[2021-03-18 06:28:40.1616029120 GMT] INFO::<module>():: Number of validation samples : 4496
[2021-03-18 06:28:40.1616029120 GMT] INFO::<module>():: Begining of training k : 1 fold for 10 epochs.

Killed
[richard@master two_way_translator_bilstm_with_battn_test8]$
```

Note:

1. In the above example, a python program 'app.py' is executed for training a deep learning model using the singularity exec command using the **--nv** option for using the gpus and the execution is done from the OS terminal (not the Singularity Shell), also the absolute path of the container needs to be specified, for example '/home/richard/local_sty_dkr_dlgpu'.
2. Please note that, this method can be used to test whether your application works, do not use it for long running jobs, as the scheduler kills any non-pbs jobs (refer the second screenshot given above).
3. Singularity commands can be used inside a shell script along with PBS command and submitted to the PBS job queues using the **qsub** command (for long running jobs) as shown in the following sections in this manual.

3. Configuring, Submitting and Managing PBS jobs

This section provides the basic knowledge to get started with PBS jobs, please note that only the bare minimum configuration required to run jobs is provided in this manual, for exploring any other features of the PBS 20.0.0 community edition, please refer to the links provided in the General Notes section of this manual.

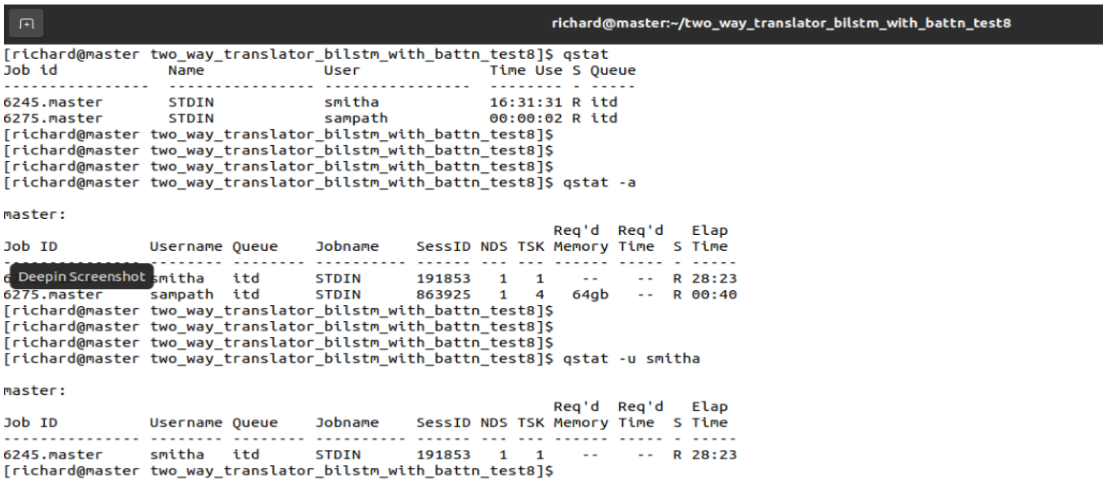
3.1 Viewing queue status

The default queue for the PBS system is named '**itd**', all jobs submitted go through this queue. The status of the jobs and other details can be accessed using the **qstat (queue status)** command.

Command: qstat [–a | -u <username>]

Notes: -a option displays all jobs on the queue, while –u with username gives a specific user's jobs. [**Job Status: R** – for running, **Q** – for jobs waiting for resources or their turn, **E** – for jobs terminated due to error, **H** – for jobs put on hold by the user, admin or by the system itself]

Output:



```
richard@master:~/two_way_translator_bilstm_with_battn_test8
[richard@master two_way_translator_bilstm_with_battn_test8]$ qstat
Job id            Name             User              Time Use S Queue
----------------  ---------------- ----------------  -------- - -----
6245.master       STDIN            smitha            16:31:31 R itd
6275.master       STDIN            sampath           00:00:02 R itd
[richard@master two_way_translator_bilstm_with_battn_test8]$
[richard@master two_way_translator_bilstm_with_battn_test8]$
[richard@master two_way_translator_bilstm_with_battn_test8]$
[richard@master two_way_translator_bilstm_with_battn_test8]$ qstat -a

master:
                                                            Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID NDS TSK Memory Time  S Time
--------------- -------- -------- ---------- ------ --- --- ------ ----- - -----
6 Deepin Screenshot mitha itd     STDIN      191853   1   1    --    --  R 28:23
6275.master     sampath  itd      STDIN      863925   1   4   64gb   --  R 00:40
[richard@master two_way_translator_bilstm_with_battn_test8]$
[richard@master two_way_translator_bilstm_with_battn_test8]$
[richard@master two_way_translator_bilstm_with_battn_test8]$
[richard@master two_way_translator_bilstm_with_battn_test8]$ qstat -u smitha

master:
                                                            Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID NDS TSK Memory Time  S Time
--------------- -------- -------- ---------- ------ --- --- ------ ----- - -----
6245.master     smitha   itd      STDIN      191853   1   1    --    --  R 28:23
[richard@master two_way_translator_bilstm_with_battn_test8]$
```

3.2 Configuring a PBS job and submitting it to the PBS job queue

The resource list i.e. The amount of memory required, number of cpus, number of gpus and targeted host name can be specified in two ways, which are as follows:

i. **Configuring the resource list through a shell script and submitting the job:**

The following shell script is an example of how the configuration for a PBS job can be done for running a python application.

{NOTE: PBS commands start with #PBS,

For example, #PBS –M mail@test.com is a PBS command and not a comment,

If you need to comment this line use ###, for example ###PBS –M mail@test.com}

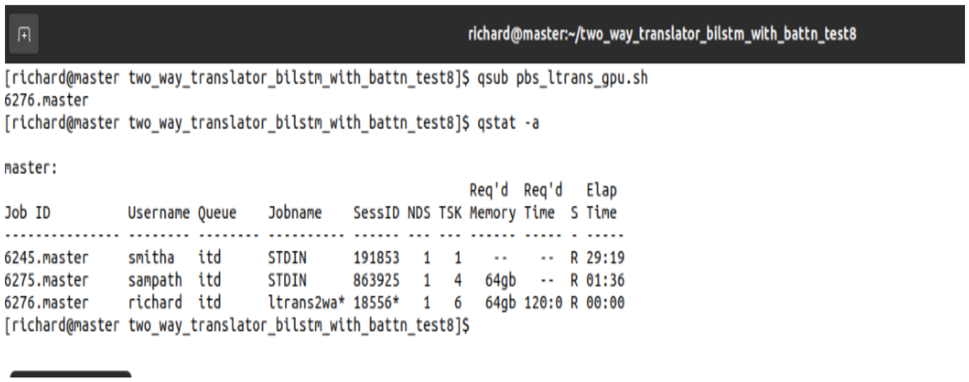
```
#!/bin/sh

### The following command will set a user-friendly job name

#PBS -N ltrans2way8

### The following command will set a user-friendly project name which is your department code by default

#PBS -P LTRANS

### The following command, specifies when email notifications can be sent: bea stand for the following : b- at begining of job execution, e-at the end of the job and a- when the job is aborted.

#PBS -m bea

### Specify your email address for notifications.

#PBS -M mail@test.com

#### This command will specify the resource list

#PBS -l select=1:mem=64G:ncpus=6:ngpus=1:host=dgx

###The following command can be used for chaining resources from 2 nodes, but is not currently effective in the system (advisable not to use this command)

###PBS -l select=1:host=master+1:mem=64G:ncpus=2:ngpus=1:host=node1

###If you are unsure about node selection, use the following command, which will assign memory, cpus and gpus from any available node.

###PBS –l select=1:mem=64G:ncpus=6:ngpus=1
```

```
### Specify "wallclock time" required for this job, hhh:mm:ss, the job will be terminated if the
running time of the job exceeds the walltime specified

#PBS -l walltime=120:00:00

###This command will create error and output files with the formats: <jobname>.o<jobid> for
output from the current shell script and <jobname>.e<jobid> for any errors while executing the
PBS commands in this shell script.

#PBS -j oe

###The following command can be used if you need to specify any project specific software
dependencies

###PBS -l software=replace_with_Your_software_name

# After job starts, must go to working directory.

### $PBS_O_WORKDIR is the directory from where the job is fired.

#$PBS_O_WORKDIR="/home/richard/two_way_translator_bilstm_with_battn_test8"

echo "==============================="

echo $PBS_JOBID

#cat $PBS_NODEFILE

echo "==============================="

cd $PBS_O_WORKDIR

#job

###The following command can be used for mpi programs ie CUDA based parallel programming

###time -p mpirun -n {n*m} executable

###If singularity is used then uncomment the following line

###singularity exec --nv /home/richard/local_dkr_ubuntu python3 app.py

### The following command is the target application for the PBS job

python3 app.py

###NOTE

### The job line is an example: users need to change it to suit their applications

### The PBS select statement picks n nodes each having m free processors

### OpenMPI needs more options such as $PBS_NODEFILE
```

NOTE: Please make the shell script executable using **chmod +x script_name.sh**

Command: qsub script_name.sh

Output:



```
richard@master:~/two_way_translator_bilstm_with_battn_test8
[richard@master two_way_translator_bilstm_with_battn_test8]$ qsub pbs_ltrans_gpu.sh
6276.master
[richard@master two_way_translator_bilstm_with_battn_test8]$ qstat -a

master:
                                                            Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID NDS TSK Memory Time  S Time
--------------- -------- -------- ---------- ------ --- --- ------ ----- - -----
6245.master     smitha   itd      STDIN      191853   1   1    --    -- R 29:19
6275.master     sampath  itd      STDIN      863925   1   4   64gb   -- R 01:36
6276.master     richard  itd      ltrans2wa* 18556*   1   6   64gb 120:0 R 00:00
[richard@master two_way_translator_bilstm_with_battn_test8]$
```

ii. **Configuring the resource list and submitting the job directly through the qsub command:**

Alternatively, the same PBS job specified in the shell script above can be run using the qsub command as follows

Command: qsub [-option_name specification]

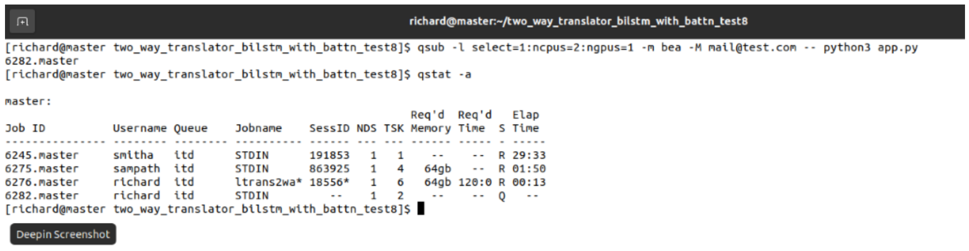
Ex: qsub –l select=1:mem=64G:ncpus=6:ngpus=1:host=dgx -- python3 app.py

The above command specifies resource list to select 1 node with 64GB memory, number of CPUs=6, number of GPUs=1 and the node name should be dgx.

Note: Other options can be specified such as email, for example:

qsub -l select=1:ncpus=2:ngpus=1 -m bea -M mail@test.com -- python3 app.py

Output:



```
richard@master:~/two_way_translator_bilstm_with_battn_test8
[richard@master two_way_translator_bilstm_with_battn_test8]$ qsub -l select=1:ncpus=2:ngpus=1 -m bea -M mail@test.com -- python3 app.py
6282.master
[richard@master two_way_translator_bilstm_with_battn_test8]$ qstat -a

master:
                                                            Req'd  Req'd   Elap
Job ID          Username Queue    Jobname    SessID NDS TSK Memory Time  S Time
--------------- -------- -------- ---------- ------ --- --- ------ ----- - -----
6245.master     smitha   itd      STDIN      191853   1   1    --    -- R 29:33
6275.master     sampath  itd      STDIN      863925   1   4   64gb   -- R 01:50
6276.master     richard  itd      ltrans2wa* 18556*   1   6   64gb 120:0 R 00:13
6282.master     richard  itd      STDIN         --    1   2    --    -- Q   --
[richard@master two_way_translator_bilstm_with_battn_test8]$
```

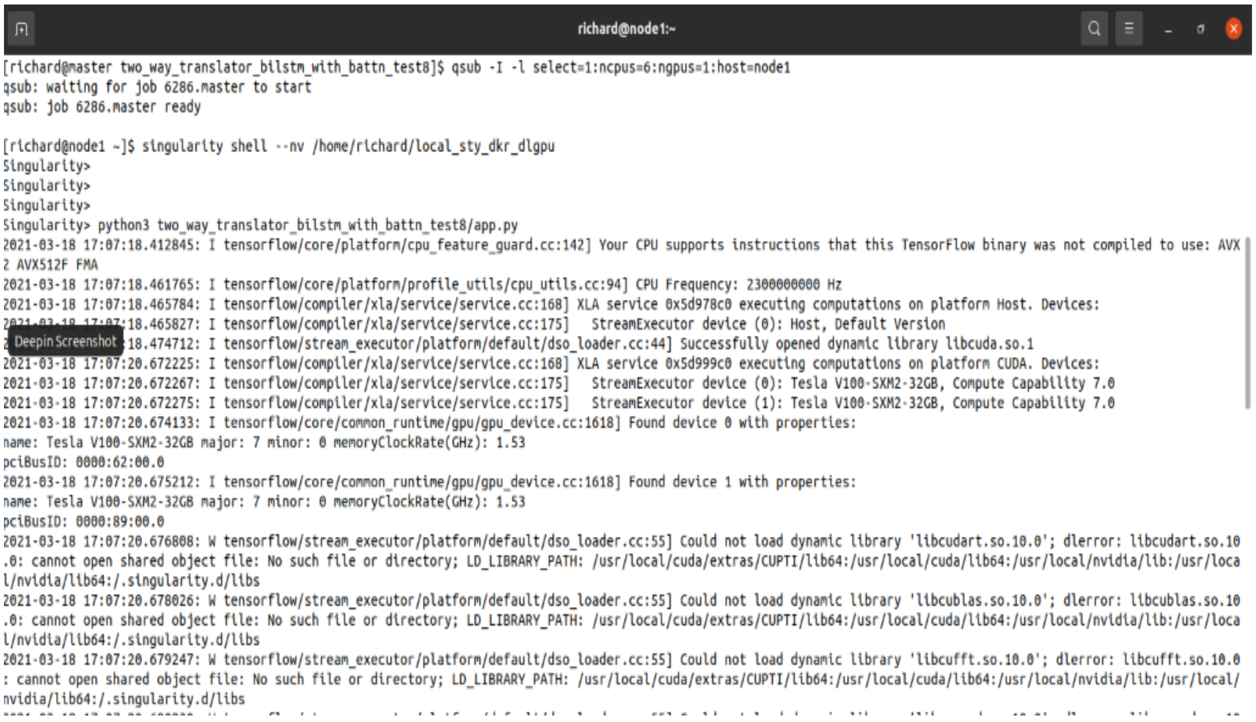
iii. Configuring the resource list and submitting a non-interactive singularity container-based job directly through the qsub command:

In certain scenarios, an interactive job might be needed, in such a case use –I with gsub before specifying any other details such as resource list and commands/scripts. For example, if you wish to run a python application in a Singularity shell in an interactive mode for the singularity container created in section 2, then use the following steps:

a. qsub –I –l select=1:ncpus=6:ngpus=1:host=node1
b. singularity shell --nv /home/richard/local_sty_dkr_dlgpu/
c. Singularity-prompt> python3 /home/richard/two_way_translator_with_battn_test8/app.py

In the above example, an interactive job is started with the specified resources needed, following which the shell terminal of the singularity container is opened and then the python application is run. The screenshot given below shows the output of the above steps.

Output:



```
[richard@master two_way_translator_bilstm_with_battn_test8]$ qsub -I -l select=1:ncpus=6:ngpus=1:host=node1
qsub: waiting for job 6286.master to start
qsub: job 6286.master ready

[richard@node1 ~]$ singularity shell --nv /home/richard/local_sty_dkr_dlgpu
Singularity>
Singularity>
Singularity>
Singularity> python3 two_way_translator_bilstm_with_battn_test8/app.py
2021-03-18 17:07:18.412845: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX
2 AVX512F FMA
2021-03-18 17:07:18.461765: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 2300000000 Hz
2021-03-18 17:07:18.465784: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x5d978c0 executing computations on platform Host. Devices:
2021-03-18 17:07:18.465827: I tensorflow/compiler/xla/service/service.cc:175]   StreamExecutor device (0): Host, Default Version
2          18.474712: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library libcuda.so.1
2021-03-18 17:07:20.672225: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x5d999c0 executing computations on platform CUDA. Devices:
2021-03-18 17:07:20.672267: I tensorflow/compiler/xla/service/service.cc:175]   StreamExecutor device (0): Tesla V100-SXM2-32GB, Compute Capability 7.0
2021-03-18 17:07:20.672275: I tensorflow/compiler/xla/service/service.cc:175]   StreamExecutor device (1): Tesla V100-SXM2-32GB, Compute Capability 7.0
2021-03-18 17:07:20.674133: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: Tesla V100-SXM2-32GB major: 7 minor: 0 memoryClockRate(GHz): 1.53
pciBusID: 0000:62:00.0
2021-03-18 17:07:20.675212: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 1 with properties:
name: Tesla V100-SXM2-32GB major: 7 minor: 0 memoryClockRate(GHz): 1.53
pciBusID: 0000:89:00.0
2021-03-18 17:07:20.676808: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcudart.so.10.0'; dlerror: libcudart.so.10
.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/loca
l/nvidia/lib64:/.singularity.d/libs
2021-03-18 17:07:20.678026: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcublas.so.10.0'; dlerror: libcublas.so.10
.0: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/loca
l/nvidia/lib64:/.singularity.d/libs
2021-03-18 17:07:20.679247: W tensorflow/stream_executor/platform/default/dso_loader.cc:55] Could not load dynamic library 'libcufft.so.10.0'; dlerror: libcufft.so.10.0
: cannot open shared object file: No such file or directory; LD_LIBRARY_PATH: /usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64:/usr/local/nvidia/lib:/usr/local/
nvidia/lib64:/.singularity.d/libs
```

3.3 Configuring and running a GUI based server application through PBS

Some scenarios might require GUI based server applications to be run like jupyter notebooks. The following example shows how to start a jupyter notebook, forward the port to your local machine inside the NITK network and then access the GUI of jupyter notebook on the browser of your local machine. **(NOTE: port forwarding to a machine outside the NITK network will require VPN access from CCC NITK)**

In this example the jupyter notebook is launched on the remote server using the following steps:

1. Create an interactive PBS job using the following command

   qsub –I -l select=1:ncpus=4:ngpus=1:host=master

(An interactive job with resources reserved on the master node with 4 cpus and 1 gpu will be triggered).

2. Next open the shell terminal of the Singularity container created in Section 2.2, using the command:

   singularity shell --nv /home/richard/local_sty_dkr_dlgpu/

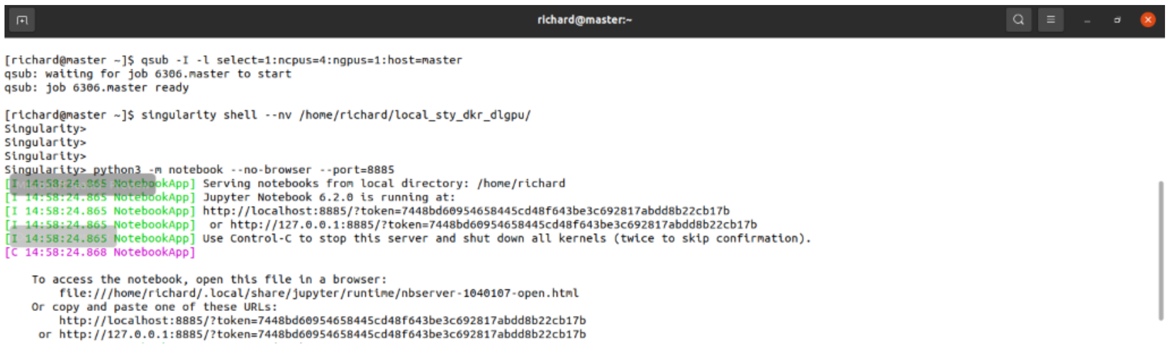
3. Finally launch the jupyter notebook in the Singularity terminal using the following command in a no browser mode on a port specified by the user (The default port is 8888)

   python3 -m notebook --no-browser --port=8885

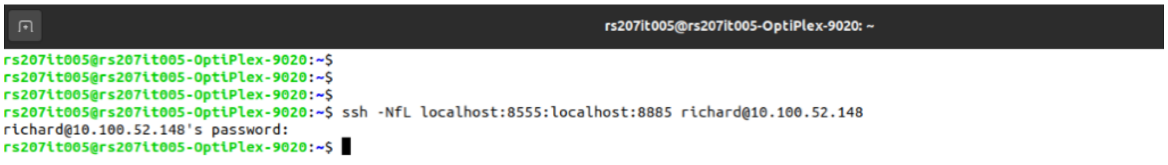
4. The above command will launch the jupyter notebook server which can be accessed on your local machine after performing the port forwarding of the application on the remote server to your local machine. The following command is one such example. (**Note**: localhost should be written as it is, unless you are specifically using a different ip other than 127.0.0.1)

ssh -NfL localhost:8555:localhost:8885 richard@10.100.52.148

5. The following screenshots illustrate the above steps:

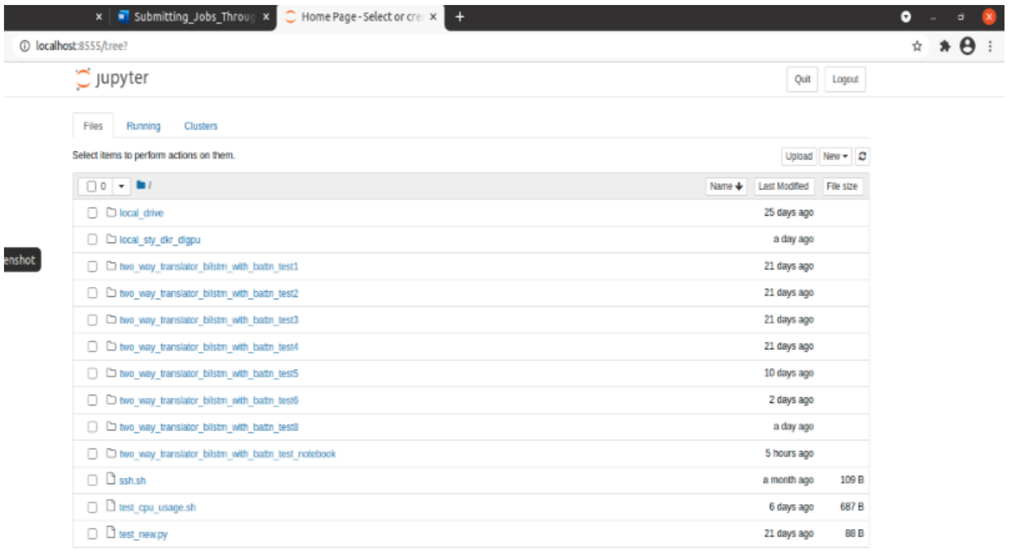


```
[richard@master ~]$ qsub -I -l select=1:ncpus=4:ngpus=1:host=master
qsub: waiting for job 6306.master to start
qsub: job 6306.master ready

[richard@master ~]$ singularity shell --nv /home/richard/local_sty_dkr_dlgpu/
Singularity>
Singularity>
Singularity>
Singularity> python3 -m notebook --no-browser --port=8885
[I 14:58:24.865 NotebookApp] Serving notebooks from local directory: /home/richard
[I 14:58:24.865 NotebookApp] Jupyter Notebook 6.2.0 is running at:
[I 14:58:24.865 NotebookApp] http://localhost:8885/?token=7448bd60954658445cd48f643be3c692817abdd8b22cb17b
[I 14:58:24.865 NotebookApp]  or http://127.0.0.1:8885/?token=7448bd60954658445cd48f643be3c692817abdd8b22cb17b
[I 14:58:24.865 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 14:58:24.868 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///home/richard/.local/share/jupyter/runtime/nbserver-1040107-open.html
    Or copy and paste one of these URLs:
        http://localhost:8885/?token=7448bd60954658445cd48f643be3c692817abdd8b22cb17b
     or http://127.0.0.1:8885/?token=7448bd60954658445cd48f643be3c692817abdd8b22cb17b
```

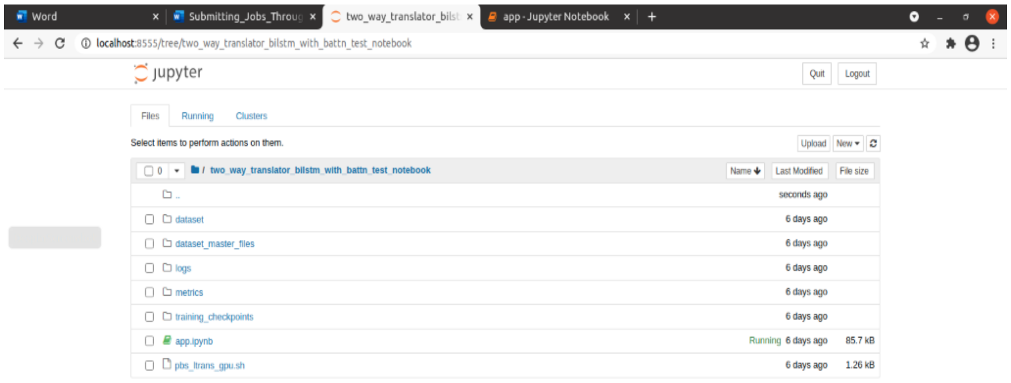


```
rs207it005@rs207it005-OptiPlex-9020:~$
rs207it005@rs207it005-OptiPlex-9020:~$
rs207it005@rs207it005-OptiPlex-9020:~$
rs207it005@rs207it005-OptiPlex-9020:~$ ssh -NfL localhost:8555:localhost:8885 richard@10.100.52.148
richard@10.100.52.148's password:
rs207it005@rs207it005-OptiPlex-9020:~$ 
```

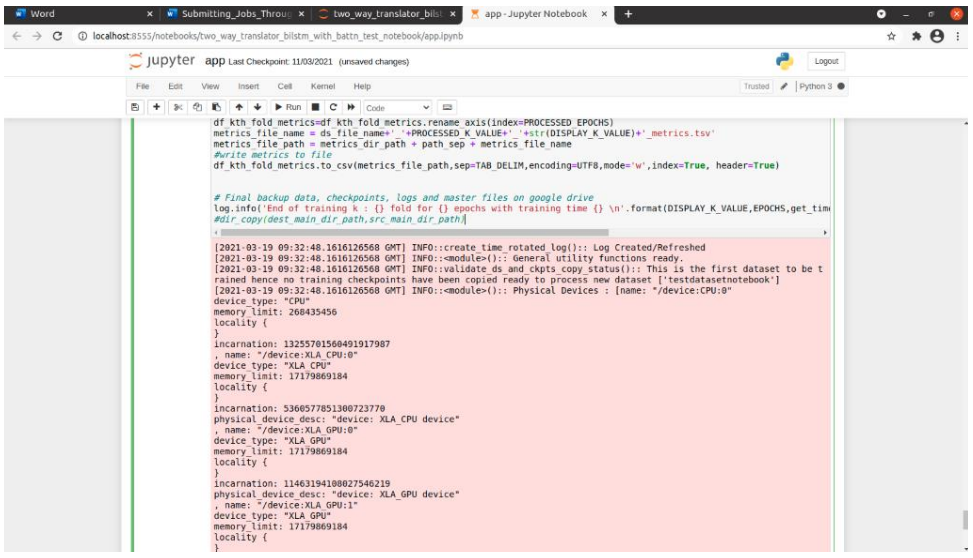
6. Finally, the jupyter notebook can be run from the browser of thelocal machine using the localhost:8555 address configured in this example. The following screenshots illustrate the entire process.



```
localhost:8555/tree?
Select items to perform actions on them.
/                                                   Name  Last Modified  File size
local_drive                                               25 days ago
local_sty_dkr_dlgpu                                       a day ago
two_way_translator_bilstm_with_battn_test1                21 days ago
two_way_translator_bilstm_with_battn_test2                21 days ago
two_way_translator_bilstm_with_battn_test3                21 days ago
two_way_translator_bilstm_with_battn_test4                21 days ago
two_way_translator_bilstm_with_battn_test5                10 days ago
two_way_translator_bilstm_with_battn_test6                2 days ago
two_way_translator_bilstm_with_battn_test8                a day ago
two_way_translator_bilstm_with_battn_test_notebook        5 hours ago
ssh.sh                                                    a month ago    109 B
test_cpu_usage.sh                                         6 days ago     687 B
test_new.py                                               21 days ago    88 B
```



```
localhost:8555/tree/two_way_translator_bilstm_with_battn_test_notebook
Select items to perform actions on them.
/ two_way_translator_bilstm_with_battn_test_notebook        Name  Last Modified  File size
..                                                                seconds ago
dataset                                                           6 days ago
dataset_master_files                                              6 days ago
logs                                                              6 days ago
metrics                                                           6 days ago
training_checkpoints                                              6 days ago
app.ipynb                                               Running   6 days ago     85.7 kB
pbs_ltrans_gpu.sh                                                 6 days ago     1.26 kB
```


Word
Submitting_Jobs_Throug
two_way_translator_bilst
app - Jupyter Notebook
localhost:8555/notebooks/two_way_translator_bilstm_with_battn_test_notebook/app.ipynb
jupyter app Last Checkpoint: 11/03/2021 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code

```
df_kth_fold_metrics=df_kth_fold_metrics.rename_axis(index=PROCESSED_EPOCHS)
metrics_file_name = ds_file_name+'_'+PROCESSED_K_VALUE+'_'+str(DISPLAY_K_VALUE)+'_metrics.tsv'
metrics_file_path = metrics_dir_path + path_sep + metrics_file_name
#write metrics to file
df_kth_fold_metrics.to_csv(metrics_file_path,sep=TAB_DELIM,encoding=UTF8,mode='w',index=True, header=True)


# Final backup data, checkpoints, logs and master files on google drive
log.info('End of training k : {} fold for {} epochs with training time {} \n'.format(DISPLAY_K_VALUE,EPOCHS,get_tim
#dir_copy(dest_main_dir_path,src_main_dir_path)
```

```
[2021-03-19 09:32:48.1616126568 GMT] INFO::create_time_rotated_log():: Log Created/Refreshed
[2021-03-19 09:32:48.1616126568 GMT] INFO::<module>():: General utility functions ready.
[2021-03-19 09:32:48.1616126568 GMT] INFO::validate_ds_and_ckpts_copy_status():: This is the first dataset to be t
rained hence no training checkpoints have been copied ready to process new dataset ['testdatasetnotebook']
[2021-03-19 09:32:48.1616126568 GMT] INFO::<module>():: Physical Devices : [name: "/device:CPU:0"
device_type: "CPU"
memory_limit: 268435456
locality {
}
incarnation: 13255701560491917987
, name: "/device:XLA_CPU:0"
device_type: "XLA_CPU"
memory_limit: 17179869184
locality {
}
incarnation: 5360577851300723770
physical_device_desc: "device: XLA_CPU device"
, name: "/device:XLA_GPU:0"
device_type: "XLA_GPU"
memory_limit: 17179869184
locality {
}
incarnation: 11463194108027546219
physical_device_desc: "device: XLA_GPU device"
, name: "/device:XLA_GPU:1"
device_type: "XLA_GPU"
memory_limit: 17179869184
locality {
}
```

7. After the job is completed, you can either quit the jupyter notebook or shut down the jupyter notebook server from the terminal. (**Note**: Please do not leave the jupyter notebooks running after the job is done as the ports will be blocked till the remote server is restarted and this will inconvenience other users.)

3.4 Deleting jobs on the PBS job queue

A user can delete only the jobs created by oneself and not the jobs created by others. The command to list all the jobs on the queue is **qstat -a** while the command to delete a job is

Command: qdel <jobid> **Ex:** qdel 6082

Sometimes a job may linger in the queue and not terminate as some processes from the job will still be accessing some of the resources, in this case, use the following command.

Command: qdel -W force <jobid> **Ex:** qdel -W force 6082

3.5 Debugging and Logs

The following process is a guide to debug the reasons for PBS job failures.

1. After a job has been submitted using qsub, its **status** in the queue can be accessed using **qstat -a**.
2. If the job status in the queue, is anything other than R (running), some more information about the job can be gathered using the commands
   i. tracejob <jobid> {This command can be used for jobs that have been executed in the previous 24 hours}
   ii. qstat –f <jobid> {This command can only be used to trace the status of the jobs visible in the output of the qstat -a command}
3. If the above two commands do not help then you can search the pbs job configuration script output and error logs described in section 2 {**Note:** These are only generated if the user has configured them in the shell script, and the **output** and **error** logs will be of the form **<jobname>.o<jobid>** and **<jobname>.e<jobid>** respectively, for example, **ltrans.o1234** and **ltrans.e1234**
4. A final resort for tracing job failures due to the PBS system can be found in the dated log at **/var/spool/pbs/server_logs** , the log will be of the form yyyymmdd {Ex: **20210319**}.
5. Any application (such as python app or jupyter notebook) generated output / errors need to be logged by the application itself, as this will be of immense help in the debugging process.

Parting Thoughts:

This manual is just a guide for the making the access of this system easier without having to go through the entire documentation on PBS and Singularity. Please feel free to explore the other features of PBS and Singularity using the resources given in the General Notes section in this manual. In case of any further queries, please contact the System Administrator.