Drop-in-session Interactive jobs

Session Start

What is an interactive session?

An interactive session takes you directly onto a compute node to enable loading modules and running jobs. Great for debugging and building software.

Starting an interactive session

As with most things HPC there are many ways to start you interactive session and it is dependent on what you need.

Tmux

Tmux is a terminal mulitplexer you can create and resume tmux sessions. Multiple sessions will be going through tmux so I am only covering it use with respect to interactive jobs a list of some tmux commands can be found searching for tmux cheatsheat. Key points:

- Starting a session `tmux new-session -s <session name>`
- Detach from a session `Ctrl + B, d`
- Attach `tmux attach -t <session name>`
- Kill a session `tmux kill session -s <session name>`
- list sessions `tmux ls`

I always recommend starting a tmux session if you will be running an interactive session and will show you additional benefits later on.

Login Nodes

Your tmux session is only recorded on a particular login node if you try and do `tmux ls` on another node your session will not be listed. You must therefore keep a record of the login node of this session and the `ssh` onto that node in the future. Can turn that into a script/alias in your `.bashrc`:

For script

```
touch session
echo ssh username@bask-pg0310u16a >> session
chmod 777 session
./session
```

alias we do not recommend changing your `.bashrc` normally

```
alias session='ssh username@baskpg0310u20a'
```

Do not have something that will automatically ssh you to a node.

Srun

You have created your tmux session to start your interactive session you should use the `srun` command. Much like `sbatch` there are a lot of potential options you can have, but the standard I have is:

```
srun --export=USER,HOME,PATH,TERM --account=wongj-bham-training --qos=bham --nodes=1-1 --ntasks=36 --
gres=gpu:1 --time=6:0:0 --pty /bin/bash
```

This is a 6 hour session requesting 1 GPU and 36 CPUs. You must run `srun` command with the `--pty` (pseudo-terminal) option. A list of other potential options are:

option	descriptions
`--pty /bin/bash`	this requests a `bash` shell on the compute node. The `--pty` option must be given at the end of the command
`--export = USER,HOME,PATH,TERM`	This exports a required subset of environment variables
`--time`	Time request of interactive job
`--qos=`	The QoS for this job
`--account=`	Defines the project account under which you want to run the job
`--x11`	**optional** allows X11 forwarding and GUI options

You can find more information here for srun and via our docs.

as before you can turn this into a script:

```
touch interactive
echo srun --export=USER,HOME,PATH,TERM --account=wongj-bham-training --qos=bham --nodes=1-1 --ntasks=36
--gres=gpu:1 --time=6:0:0 --pty /bin/bash >> interactive
chmod 777 interactive
./interactive
```

An interactive job is still a job so you will queue for your job to start so you can also add `--mail-user=` and `--mail-type=BEGIN`

If you wanted to have an iteractive session on the a100-80's can add it as a constraint `--constraint=a100-80`

Normally if you close the terminal you will end your interactive session except when you are in a tmux session. So if for some reason you are logged out off Baskerville you can login ssh to the login node that has your tmux session and resume your interactive session.

General Warnings:

- Try not to leave your interactive session idle, if it is doing something that you know should take hours feel free to detach and resume your tmux session.
- Tmux is not infallible and session can still end especially if there is a problem with the login node.
- Try to kill tmux sessions as soon as you are finished with them.

Example Run

1. Start Tmux session
    1. `tmux new-session -s session`
    2. memorise login node - create session file
    3. Create 2 panes - `Ctrl b, %`
2. Start interactive session on one pane
    1. Maximise pane - `Ctrl b, z`
    2. srun - `./interactive`
3. Can check queue to see job on another pane
    1. Minimise pane - `Ctrl b, z`
    2. Switch pane - `Ctrl b, <- or ->`
    3. monitor job - `squeue` or `watch -n 30 squeue`

CUDA Example

In tmux session:

1. Create 2 different panes `Ctrl b, %`
2. Start 2 different interactive sessions:
    1. One pane has an a100-40 GPU the other an a100-80 GPU
3. synchronize panes `Ctrl b, : setw synchronize-panes on`
4. `module purge; moudule load baskerville` - creates a fresh environment
5. Search for cuda modules - `module spider CUDA`
6. pick a version to load `module load CUDA/11.7.0`
7. Examine modules loaded with `module list`
8. Go to CUDA example transpose:
    1. `wget https://github.com/NVIDIA/cuda-samples/archive/refs/tags/v11.6.tar.gz`
    2. `tar zxf v11.6.tar.gz`
    3. `cd cuda-samples-11.6/Samples/6_Performance/transpose`
    4. `make`
9. Run example `./transpose`
10. You can edit `transpose.cu` for example `define NUM_REPS` from 1000 to 100000
11. Examine work on GPUs using `nvidia-smi`
    1. `watch -n 2 nvidia-smi`
    2. `nvidia-smi pmon` - Process Monitoring shows:
        1. SM - Streaming Multiprocessor
        2. Memory
        3. Encoder
        4. Decoder

PyTorch test

Example for this test is to use a test for PyTorch and check GPU availability and run a 10 epoch training on CIFAR10 dataset.

1. In interactive session create your venv
    1. Load the relevant modules, Python and PyTorch
    2. Create python virtual environment and pip install relevant modules:
        1. `python -m venv venv` - create virtual environment called venv
        2. `source venv/bin/activate`
        3. `pip install --upgrade pip`
        4. `pip install test-pytorch-gpu`
    3. Run `run_cifar` whilst having `nvidia-smi` on another tmux pane to see GPU use.