How to run Machine Learning programs on midway cluster

Igor Yakushin

April 2, 2021

- A reference installation of all the packages used in the class is provided on midway2 if you have trouble running on your laptop.
- I am assuming you are familiar with Linux and took my Linux video lessons which, besides Linux, also describe how to use RCC resources.

1 Running interactively with jupyter

1.1 On login node

1.1.1 When NOT to use login node

- You should not run anything heavy that requires a lot of time and computing resources: login node is shared potentially among several thousand users and its primary function is to submit jobs to compute nodes.
- Login nodes do not have GPUs.

1.1.2 When to use login node

- Running small examples shown in class is usually OK unless they require GPU.
- If you need to download data or install some program with, for example, 'conda install' or 'pip install', you have to run it on login node since compute nodes cannot do it for security reasons.

1.1.3 How to run jupyter on login node

- Connect to login node using ssh: `ssh YourUserName@midway2.rcc.uchicago.edu`
- Set up the environment for my python environmet:
`source /project2/msca/ivy2/software2/etc/MSCA_31009-4_env.sh`

- The environment defines several aliases to start jupyter. On login node, use
`jnl`.
- Copy and paste into your browser the URL, printed to screen, that starts with
`http://midway2-...`
- When in jupyter, select `MSCA_31009` kernel.

1.2 On compute node

1.2.1 When NOT to use compute node

- Data cannot be downloaded from compute nodes.
- Programs cannot be installed from compute nodes.

1.2.2 When to use compute node

- In ideal - always, except for the above two cases.
- Certainly when your calculation takes a lot of time and resources
- When you need GPU: they are available either in dedicated `mscagpu` partition or in general purpose `gpu2` partion

1.2.3 How to run jupyter on compute node

- Use UChicago VPN, otherwise, you cannot connect the browser on your laptop to jupyter server running on the compute node.
- Connect to login node using ssh: `ssh YourUserName@midway2.rcc.uchicago.edu`
- Request a compute node for interactive use, for example:
`sinteractive -p broadwl --time=03:00:00`
- The above will give you one node from `broadwl` partition for 3 hours with default resources: 1 node, 1 CPU core, 1.9G of RAM
- There are options to ask for more resources. For example:
  - `--exclusive` would give you all CPU cores (28) and all the memory (about 58G) in the node
  - `--mem=10G` would give you 10G of RAM
  - `--ntasks=8` would give you 8 CPU cores. If you are using this option, you should also set `OMP_NUM_THREADS` accordingly before launching jupyter to restrict how many threads would numpy use:
    `export OMP_NUM_THREADS=8`.
    Othewise, by default numpy would try to use 28 threads which would

slow down a program since only 8 hardware cores are available and a lot of time would be wasted on context switching.

- Here is some information about midway2's partions:
  - midway2 cluster consists of about 500 compute nodes
  - The compute nodes are divided into partitions depending on node properties: `broadwl` - general purpose CPU nodes, `mscagpu` - GPU nodes for MSCA, `gpu2` - general purpose GPU nodes, `bigmem2` - big memory nodes, etc.
  - For the first half of the course you would not need GPU and can use `broadwl`
  - When we get to GPU accelerated Deep Learning programs, you might have to use either `mscagpu` or `gpu2` partitions.
- `sinteractive` will eventually bring you to a compute node that you can use interactively
  - How long it takes to acquire a compute node, depends on how busy the corresponding partition is
  - You can find how busy a particular parition is with `sinfo`. For example:
    `sinfo -p broadwl`
    - `idle` state means that the node is completely idle
    - `mix` state means that the node is partially occupied and depending on how much resources you ask for, you might or might not be able to run on it
    - To find to what extent a particular node is busy, you can use `scontrol` command. For example,
      `scontrol show node midway-bigmem03`
- Once you get to the compute node, set up the environment:
  `source /project2/msca/ivy2/software2/etc/MSCA_31009-4_env.sh`
- The environment defines several aliases to start jupyter. On compute node, use
  `jnc`.
- Copy and paste into your browser the URL, printed to screen, that starts with
  `http://10...`
- When in jupyter, select `MSCA_31009` kernel.

1.3 Pros of using jupyter

- Easier to develop a program
- Plots and markdown are together with the code
- Can save as html, pdf, etc

1.4 Cons of using jupyter

- You have to stay connected all the time while your program is running. For the programs that run for many hours or days, this is unacceptable. It is also problematic if you have bad Internet connection.
- You have to wait for sinteractive to give you a node which sometimes might take hours depending on how busy the cluster is.
- Poor resource utilization: when using jupyter the computer is mostly idle while you are thinking what to do or taking a coffee break. On cloud you would pay extra for scratching your head in jupyter, on midway - less users can utilize the cluster.
- If cons outweight pros, submit jobs in batch.

2 Running in batch

2.1 Pros of running in batch

- It takes a few seconds to submit your job to the scheduler. After that you can shutdown your laptop.
- The scheduler will run your job when it can.
- No compute time is wasted on thinking what to do.
- Your program can run for several days, you do not need to stay connected.

2.2 Cons of running in batch

- Not interactive.
- Need to program to save pictures.
- Your program should already be more or less close to production state.

2.3 How to run jobs in batch

- Convert your program to python, get rid of any jupyter specific things. If you developed your program in jupyter, you can save it as python.
- Once you convert to python and get rid of any jupyter specifics, it should be possible to run your program, let's call it `hello.py`, as follows:
  `python hello.py`
- Create a shell script, let us call it `hello.batch`, that asks the scheduler for resources, sets up the environment and runs your program. For example:

```
#!/bin/bash

#SBATCH --job-name=hello
#SBATCH --output=%j_out.log
#SBATCH --error=%j_err.log
#SBATCH --time=3:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=11
#SBATCH --partition=broadwl
#SBATCH --mem=15G

# Set up the environment

export OPENMP_NUM_THREADS=11
source /project2/msca/ivy2/software2/etc/MSCA_31009-4_env.sh

# Run the program

python hello.py
```

- Above `#SBATCH` lines ask for resources and should be on top of the program before anything else.
- The script asks for 1 node, 11 CPU cores, 15G of RAM for 3 hours in broadwl partion.
- To learn for how much time you can ask in each partition, run
  `rcchelp qos`
- Submit the script to the scheduler with
  `sbatch hello.batch`
- This command returns `jobid` that you can use to query the status of your job
  `squeue -j jobid`

- Or you can query the status of all your jobs
  `squeue -u $USER`
- You can kill your job with
  `scancel jobid`
- For more details, see my Linux video lessons and RCC User Guide
  `https://rcc.uchicago.edu/docs/running-jobs/index.html`

3 midway3

- RCC has just launched a new cluster - midway3 - that you can use as well.
- At the moment you are on your own there: no reference installation is provided but you can install your own.
- To connect there, run:
  `ssh YourUserName@midway3.rcc.uchicago.edu`
- midway3 does not share so far any file systems with midway2 and MSCA does not have a dedicated project space there. You can only use `$HOME` and `/scratch/midway3/$USER`.
- MSCA does not have a dedicated partition there as well, you can use general purpose partitions.