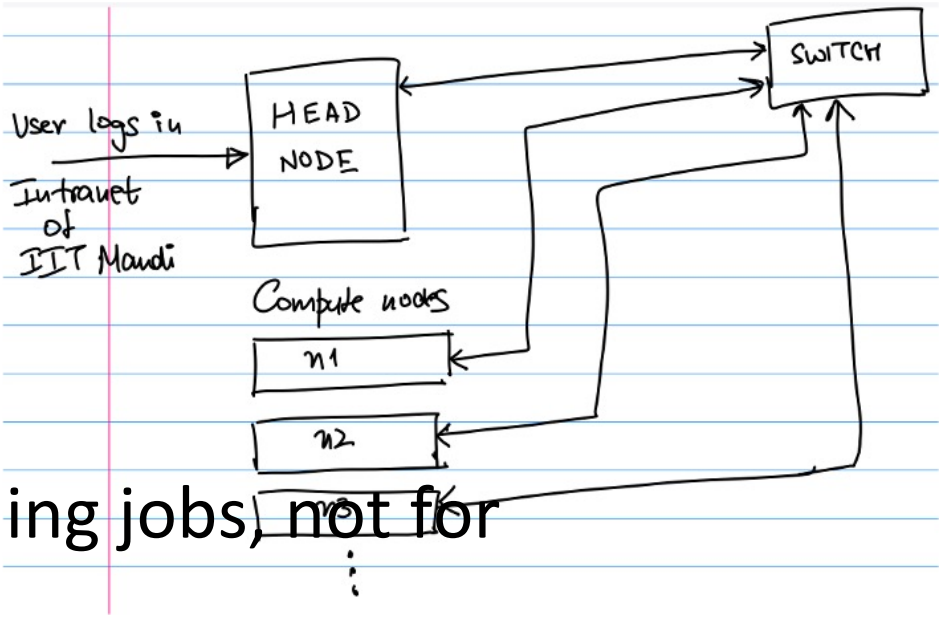
This lecture:

- HPC introduction
- Python on HPC
- Hands-on
- Exercise

HPC - Introduction



HPC topology

Head node:
Suited to login and scheduling jobs, not for computation

Compute nodes:
Suited for heavy computation, typically has 2 processors per node (machine), each processor has up to 12 cores, 64+GB RAM, 2TB local storage

HPC IIT Mandi (website tour)

- https://sites.google.com/iitmandi.ac.in/hpc-iit-mandi/
- CPUHPC (10.8.1.19): CPU-based parallelism
- GPUHPC (10.8.1.20): GU-based parallelism
- 10G connectivity between nodes
- Filesystems
  - Home: 10GB
  - Working dir (wd): 2TB
- Software: basic software + *Singularity*
- Example PBS scripts
- Queue details
- Resources

Applications

Engineering

• Fluid flow and heat transfer (Open FOAM, Fluidity, ANSYS Fluent) • Solid mechanics (ANSYS)

• Deep learning (Python modules)

Physics

• Molecular dynamics (in-house codes)

Chemistry

• Computational chemistry (Gromacs)

Biology

• Gene sequencing

IIT Mandi HPC cluster: 2884 processing cores; 300+ users • 169 nodes; 2884 cores

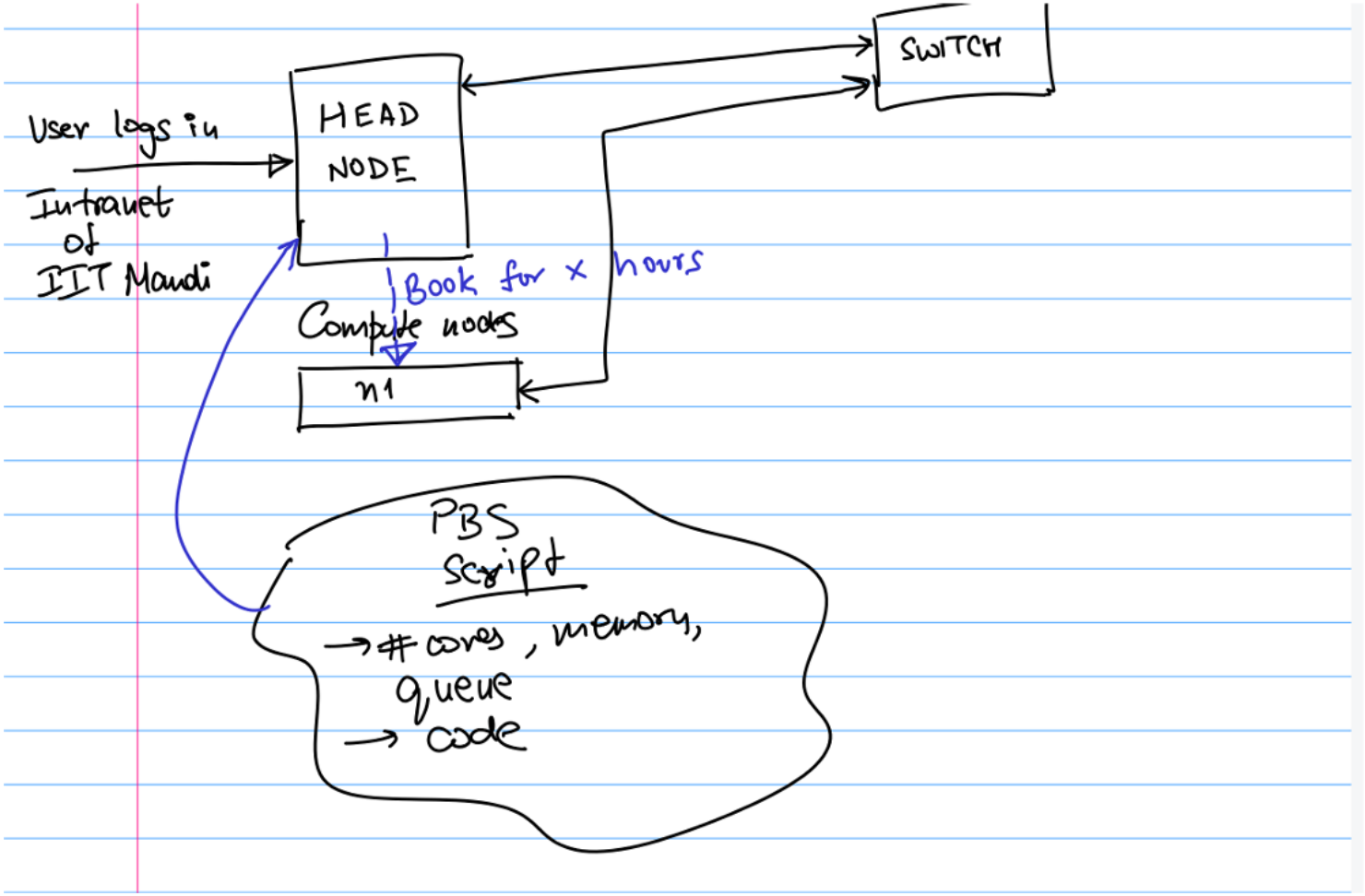
- Intel Xeon processors
- 11.5 TB memory
- 986 TB storage
- 33 Nvidia graphical processing units (GPUs) • 10Gb/s ethernet connectivity
- Cooling system

HPC – scheduler – Torque PBS

PBS: portable batch system

Queuing system for job submission which takes care of the number of jobs and excessive usage

Clusters
Filesystems


SWITCH
HEAD
NODE
User logs in
Intranet
of
IIT Mandi
Book for x hours
Compute nodes
n1
PBS
Script
→ # cores, memory,
queue
→ code

Hands-on

- Session screenshot to be recorded
- Logging in to the cluster(s) and directory structure
- Prepare virtualenv and install packages using pip
- PBS script for running a Python code
- Launching the PBS script using qsub
- Output and error files

Commands

scl —list - list all scl (Red Hat) packages available
scl enable rh-python36 bash (exit to exit from the scl)
pip3 install --user --upgrade --proxy=http://10.8.0.1:8080 pip
pip3 install --user --upgrade --proxy=http://10.8.0.1:8080 virtualenv
virtualenv ~/virtualenvs/testenv
source ./virtualenvs/testenv/bin/activate
pip3 install --upgrade --proxy=http://10.8.0.1:8080 pip
pip3 install --proxy=http://10.8.0.1:8080 numpy ipython jupyter
pip3 list - to check installed packages

qsub <pbs_script.sh>

PBS script – detailed discussion

- PBS directives used in the script
- man qsub
  - discuss environment variables
- https://docs.adaptivecomputing.com/torque/4-0-2/Content/topics/commands/qsub.htm

Qstat

qstat -an
-a - all jobs
-n - display nodes allocated to jobs
qstat -q
qstat <jobid> -f - full detail

qdel jobid
qdel all

Interactive PBS – Ipython

Qsub –I <pbs_script.sh>

Exercise 1

1. Login to the cluster
2. Copy the mysqrt.py file to wd
3. Copy a sample serial PBS script from the HPC website and edit it to calculate the square root of 2.0 with the debug mode on. The output should not be redirected into a file. See where it goes.
4. Launch the script on CPUHPC. Also, try with a different queue.
5. Launch the script on GPUHPC. Take care that the queues are different on the two clusters.
6. Also, try using the interactive mode in qsub
7. Try qstat options, and pbsnodes to check node mapping.
8. Copy the output file and error files back to your PC
9. Added challenge: can you write a bash conditional in your PBS script that runs the code only if the job is sent to n121. Use –'-l nodes=n121.cluster.iitmandi.ac.in' to test it.

Exercise 2 - advanced

- Copy the mysqrt.py module to the HPC home
- Write a python script called root.py that calculates the sqrt of number using mysqrt module and prints it. The number should be passed to the script through bash using sys.argv variable
- Write a bash loop that calls root.py for the first 1000 even numbers, i.e. 2,4,6,…
- The results should be appended into a file in the following format:
    - Sqrt of 2 is 1.414xxx, time taken xx seconds
    - Sqrt of 4 is 2.0, time taken xx seconds
- Run the bash loop over the HPC cluster in the serial queue