WATGPU USAGE SEMINAR

25/7/2024

Indy Ng (wk5ng@uwaterloo.ca)
Lucas Gamez (lmgamez@uwaterloo.ca)
Lori Paniak (ldpaniak@uwaterloo.ca)

CSCF

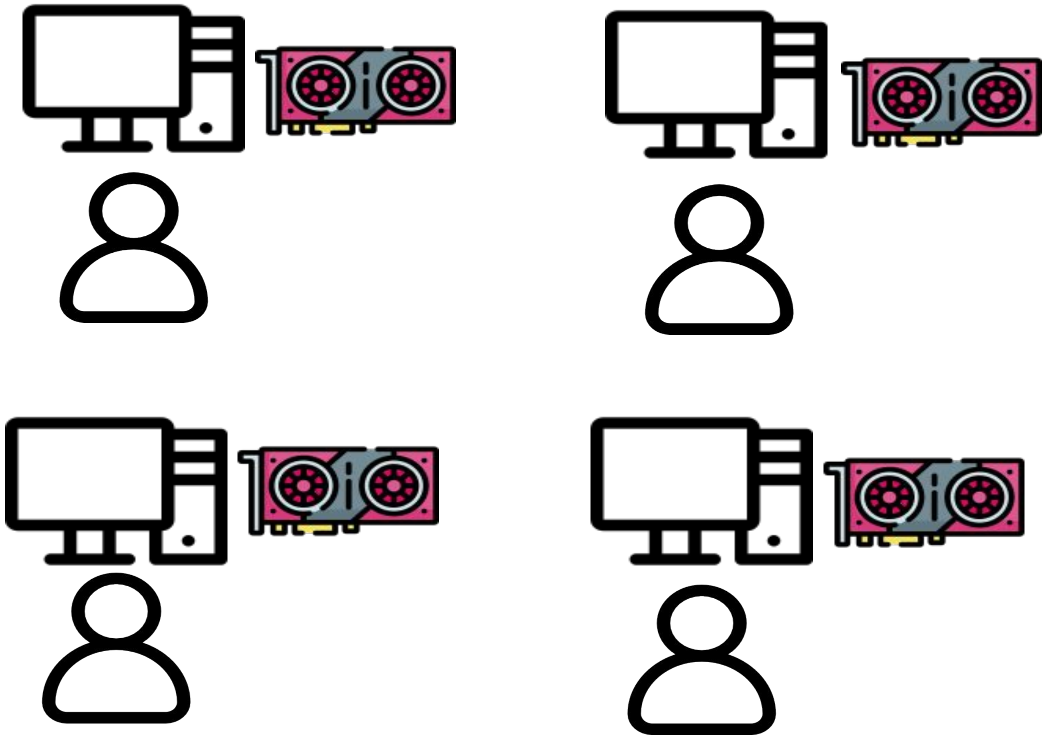
UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

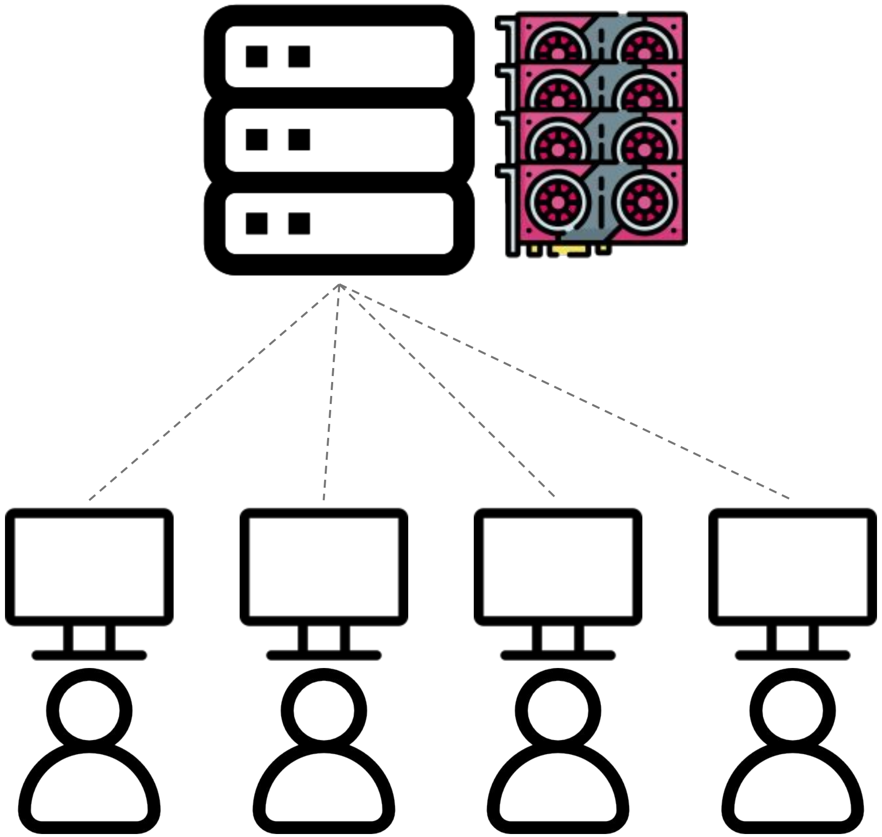
WHAT WE'LL BE COVERING TODAY...

- What is WATGPU?
- Basic usage
  - How to gain access, transferring files, storage, login vs compute nodes, partitions, monitoring your jobs, monitoring cluster resources
- Conda and Pip virtual environments on WATGPU
- Interactive and batch jobs
  - Specifying partitions, nodes, GPUs, jupyter notebooks, VS Code
- Advanced usage
  - using SIGTERM signals, checkpointing, shared directories, job arrays

WHAT IS WATGPU?

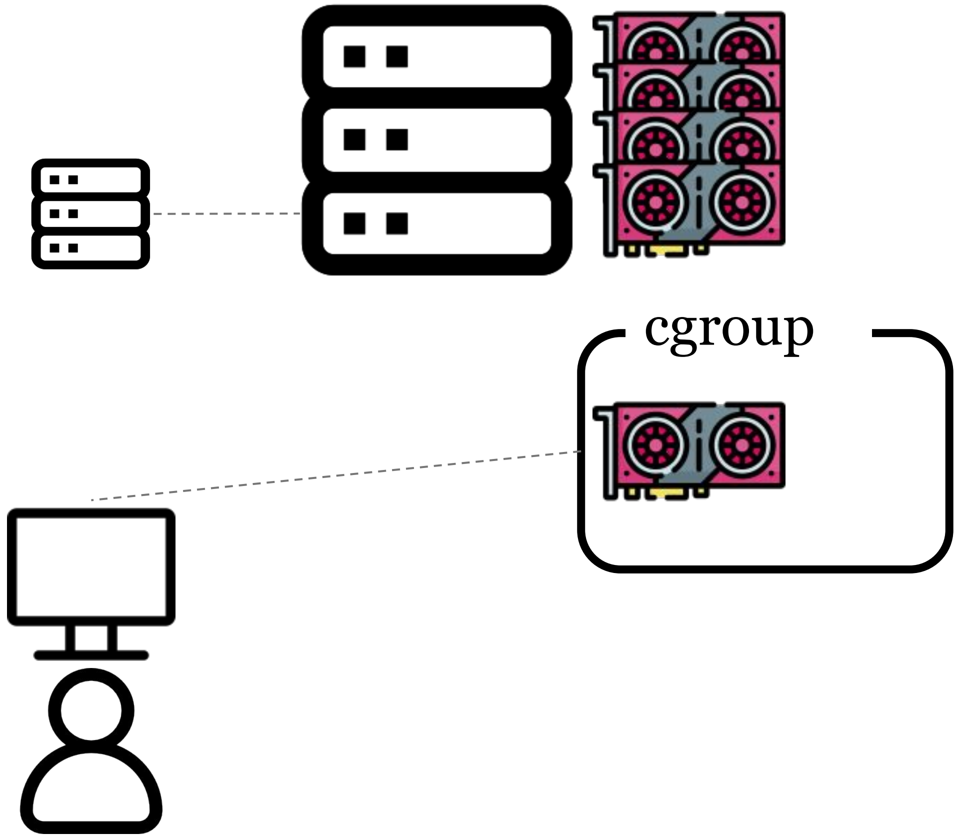
- HPC servers that host School and faculty GPUs
- Avoid idle GPUs and everyone buying expensive workstations/their own servers





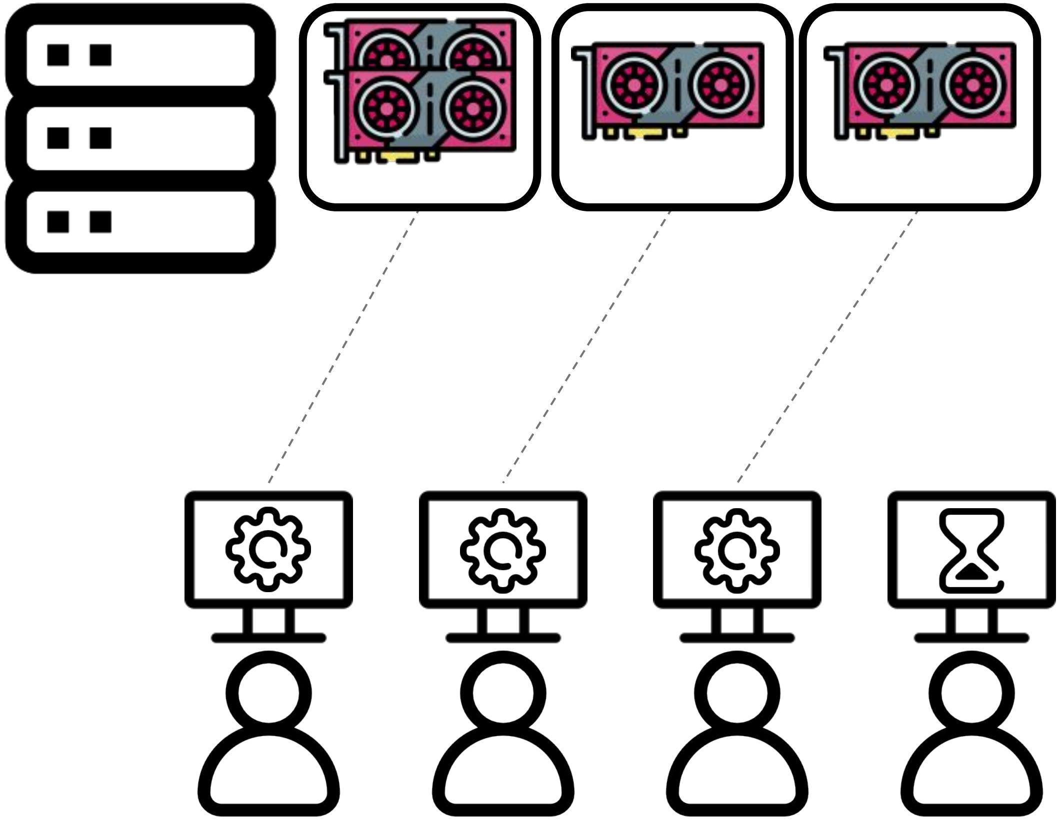
SLURM AS A RESOURCE MANAGER

- HPC cluster utilizing Slurm as a resource scheduler
- Enter **login node**
- Submit a **job**:
  - X GB of memory
  - Y GPUs
  - Z CPU cores
  - N hours/days
  - Run *<my_script>.sh*
- Get redirected to a **compute node** with confined env!



QUEUE AND PRIORITY

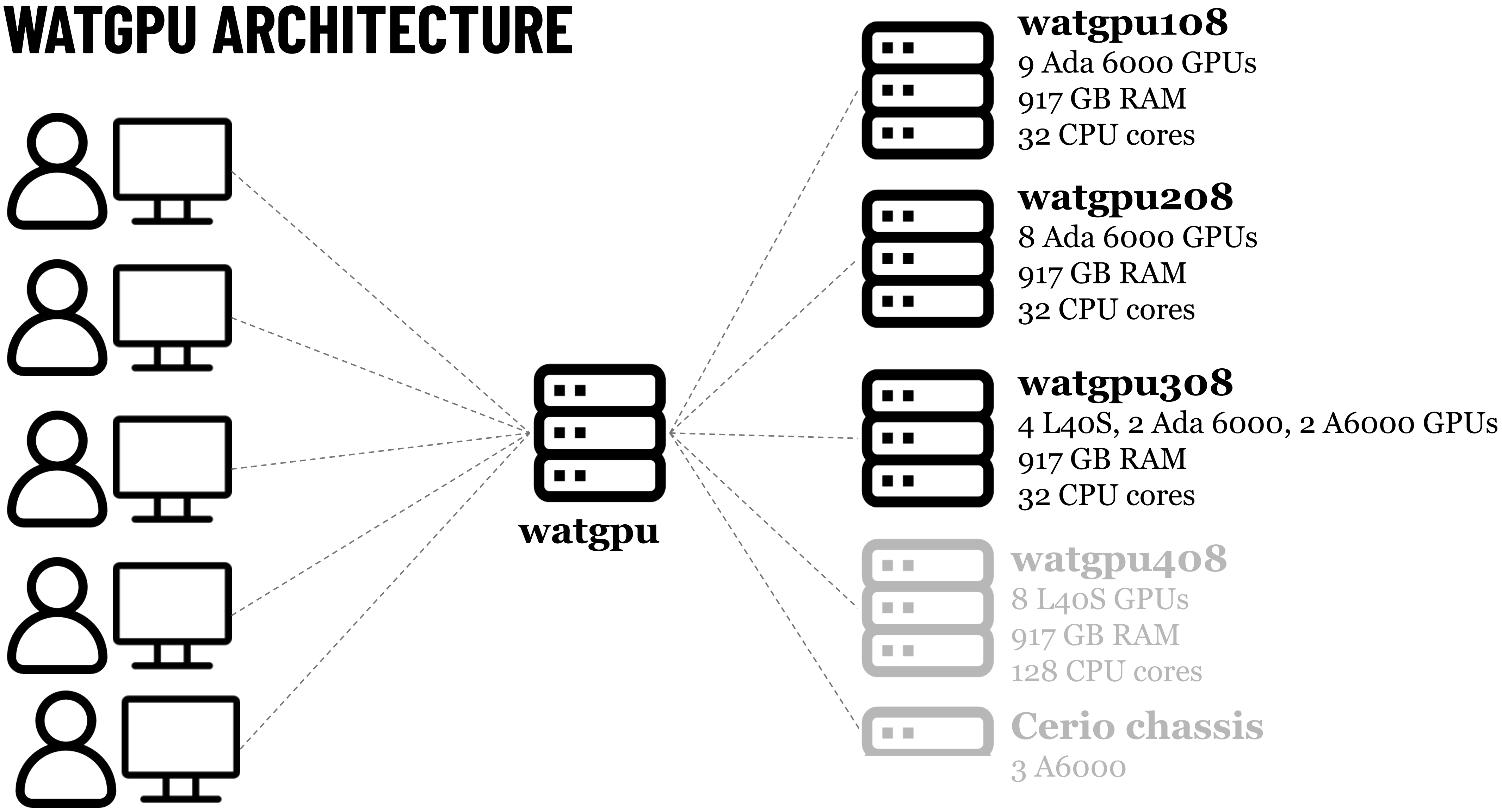
- Jobs are **queued** and executed as soon as resources are available
- Contributors gain access to priority partitions on WATGPU



Documentation:

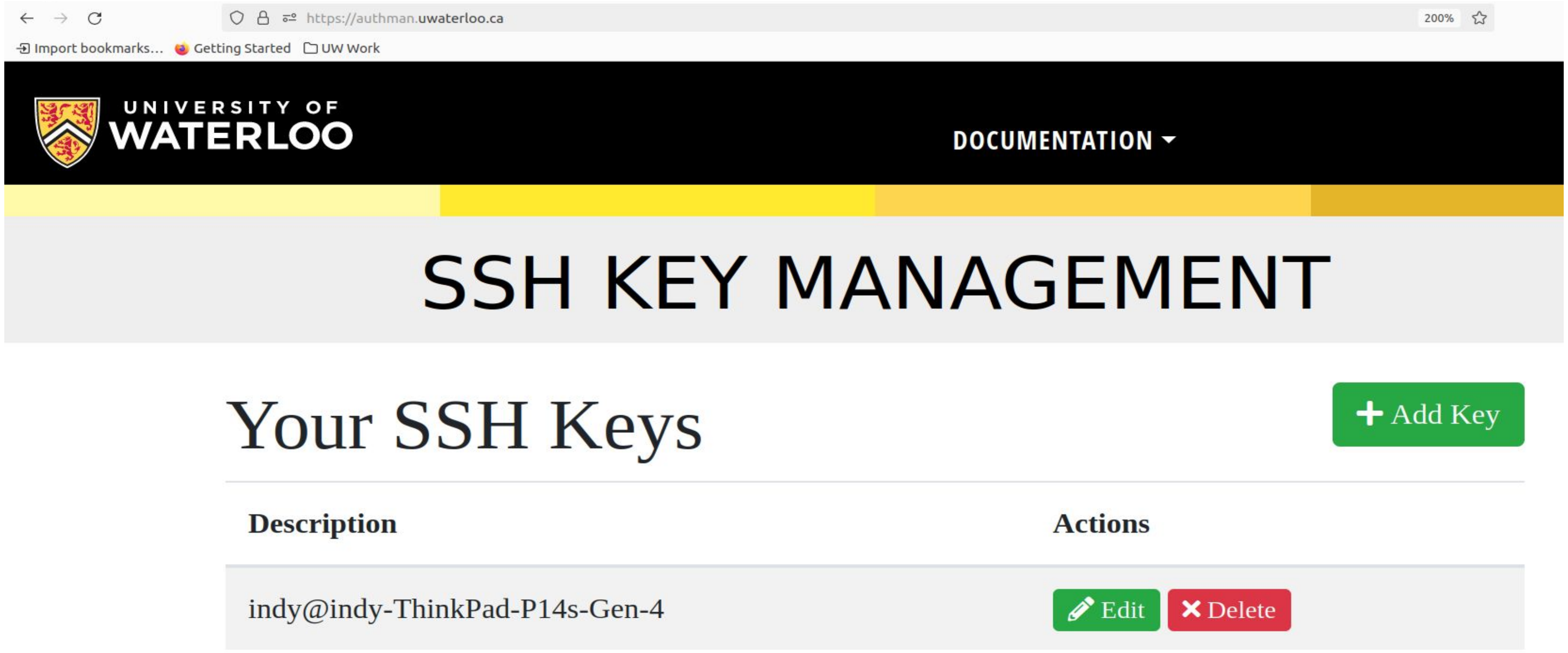
watgpu.cs.uwaterloo.ca

WATGPU ARCHITECTURE



BASIC USAGE

- How to gain access?
  - generate SSH key and upload to authman.uwaterloo.ca
  - ask your supervisor to send in a request (or CC your supervisor when you send a request to us)



BASIC QUESTIONS

- Connecting to the server

```
user@localhost:~$ ssh <userID>@watgpu.cs.uwaterloo.ca
```

- Login vs compute nodes
  - long-running user scripts/programs on login node will be killed!
- How to transfer files to and from WATGPU?
  - rsync/scp (compress before transferring) or SFTP
  - default storage of 1TB per user

BASIC QUESTIONS

- How to monitor running jobs?

```
(base) user@watgpu:~$ squeue
```

- How to monitor cluster resources?

```
(base) user@watgpu:~$ sresources
```

- How to submit jobs?
  - salloc vs sbatch

Documentation:

watgpu.cs.uwaterloo.ca/slurm.html#queues

UNIVERSITY OF WATERLOO | FACULTY OF MATHEMATICS

CONDA AND PIP VENVS ON WATGPU

- Default setup for all users is conda
  - when you SSH into WATGPU, you should see something like

```
(base) wk5ng@watgpu:~$
```

- You can create pip venvs while this base conda env is activated (accounts have Python through this base env)
- After creation, you can conda deactivate and activate your pip venv normally

```
(base) wk5ng@watgpu:~/test_job$ conda deactivate
wk5ng@watgpu:~/test_job~$ source .venv/bin/activate
(.venv) wk5ng@watgpu:~/test_job~$
```

PYTORCH/TENSORFLOW ENV SETUP

- Create a conda environment, then install PyTorch/TensorFlow according to installation guides for each package

```
(base) wk5ng@watgpu:~$ conda create -n pytorch_base
```

```
(base) wk5ng@watgpu:~~$ conda create -n tensorflow_base
```

- PyTorch:

```
(base) wk5ng@watgpu:~$ conda activate pytorch_base
(pytorch_base) wk5ng@watgpu:~$ conda install pytorch torchvision torchaudio
pytorch-cuda=12.1 -c pyttorch -c nvidia
```

- TensorFlow:

```
(base) wk5ng@watgpu:~$ conda activate tensorflow_base
(tensorflow_base) wk5ng@watgpu:~~$ pip install tensorflow[and-cuda]
```

PYTORCH/TENSORFLOW ENV SETUP

- To check PyTorch is installed properly:

```
(base) wk5ng@watgpu108:~$ conda activate pytorch_base
(pytorch_base) wk5ng@watgpu108:~$ python
Python 3.12.4 | package by Anaconda, Inc | (main, Jun 18 2024) [GCC 11.2.0] on
linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> torch.cuda.is_available()
True
>>> torch.cuda.device_count()
1
```

- To check TensorFlow is installed properly:

```
(base) wk5ng@watgpu108:~$ conda activate tensorflow_base
(tensorflow_base) wk5ng@watgpu108:~$ python3 -c “import tensorflow as tf;
print(tf.config.list_physical_devices(‘GPU’))”
[PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]
```

INTERACTIVE AND BATCH JOBS

- For sbatch/salloc, how to specify:
  - partitions (<CONTRIB>, ALL, SCHOOL)
  - time
  - GPUs, RAM, CPU cores
- VS Code

Documentation:

watgpu

```
#!/bin/bash

#SBATCH --time=1:00:00
#SBATCH --mem=4GB
#SBATCH --cpus-per-task=4
#SBATCH --gres=gpu:0

#SBATCH -o JOB%j.out # File to which STDOUT will be written
#SBATCH -e JOB%j_err.out  # File to which STDERR will be written


# Set different types of notifications:
#SBATCH --mail-user=aketchum@uwaterloo.ca
#SBATCH --mail-type=ALL


echo "Hello World"

source activate my_conda_env
python experiment_25.py
```

PARTITIONS

- Partitions are subsets of nodes and resources on the cluster
- Three types of partitions:
  - **ALL**: will run on any available resources **(default partition)**
  - **SCHOOL**: will run on school-owned resources
  - **<CONTRIB>**: will run on contributed resources (restricted to contributors)
- Contributor partitions allow priority access to contributed GPUs
- To see the list of partitions:
  - sinfo

TYPICAL SALLOC EXAMPLE

- Specify job parameters in **salloc**

```
(base) wk5ng@watgpu:~$ salloc --partition=ALL --time=00:02:00
--cpus-per-tark=4 --mem=16G --gres=gpu:1
```

- There are default values for these job parameters
- Slurm will allocate resources and you'll be SSH-ed into the allocated compute node

```
(base) wk5ng@watgpu:~$ salloc --partition=ALL --time=00:02:00
--cpus-per-tark=4 --mem=16G --gres=gpu:1
salloc: Granted job allocation 14153
salloc: Waiting for resources configuration
salloc: Nodes watgpu108 are ready for job
(base) wk5ng@watgpu:~$
```

TYPICAL SBATCH EXAMPLE

- Create a Bash script (for eg: train_XYZ.sh)
  - In the script, specify the job parameters for Slurm using the #SBATCH keyword

```
#!/bin/bash
#SBATCH --time=00:05:00
#SBATCH --mem=4GB
#SBATCH --cpus-per-task=1
#SBATCH --gres=gpu:1
#SBATCH --partition=ALL
#SBATCH --mail-user=aketchum@uwaterloo.ca
#SBATCH --mail-type=ALL
#SBATCH --job-name=pytorch-test
#SBATCH -o pytorch-test-run-%A.out
```

TYPICAL SBATCH EXAMPLE

- You can activate a specific conda/pip environment, use Bash commands, run nvidia-smi, run specific Python scripts

```
echo “Hello World”

nvidia-smi
nvidia-smi -a | grep Serial

source activate my_conda_env
which python

python experiment_25.py
```

- Submit your job using **sbatch train_XYZ.sh**

COMMON SBATCH/SALLOC COMMAND OPTIONS

Example	Comment
--time=1:00:00 --time=2-00:00:00	Max walltime for job
--mem=2G	Memory per node
--mem-per-cpu=3G	Memory per CPU *
--cpus-per-task=4 -c 4	CPU cores per task
--gres=gpu:1	GPUs per node **
--mail-user=aketchum@uwaterloo.ca	Email address for updates from Slurm
--partition=ALL --partition=SCHOOL --partition=[CONTRIBUTOR]	Specifies partition in Slurm
--job-name=”jobname”	Custom job name
--output=”JOB_%j.out”	File to which STDOUT will be written
--error=”JOB_%j_err.out”	File to which STDERR will be written

* best when memory use scales with number of CPU cores

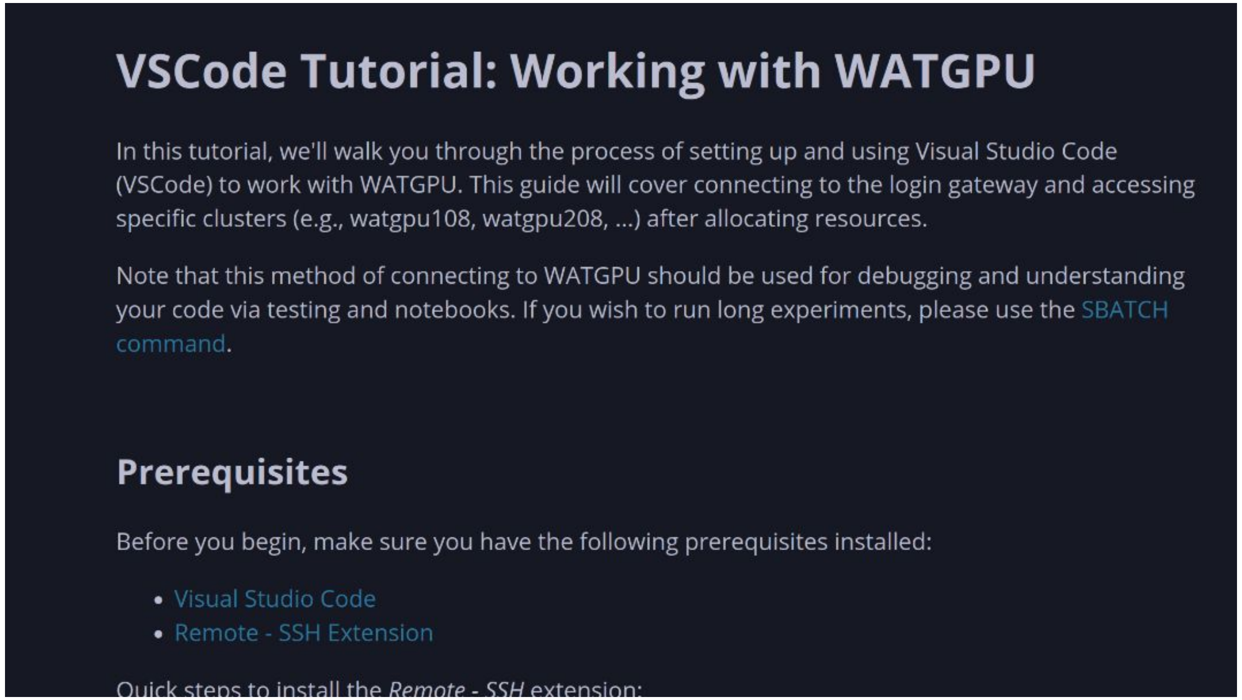
** multi-node GPU usage might require special configuration of your GPU jobs, talk to us

Slurm commands option summary

VS CODE AND WATGPU

- On WATGPU website, detailed instructions on how to integrate WATGPU and VS Code!
- Edit code, submit interactive and/or batch jobs, look at output files

watgpu.cs.uwaterloo.ca/vscode.html



VSCode Tutorial: Working with WATGPU

In this tutorial, we'll walk you through the process of setting up and using Visual Studio Code (VSCode) to work with WATGPU. This guide will cover connecting to the login gateway and accessing specific clusters (e.g., watgpu108, watgpu208, ...) after allocating resources.

Note that this method of connecting to WATGPU should be used for debugging and understanding your code via testing and notebooks. If you wish to run long experiments, please use the SBATCH command.

Prerequisites

Before you begin, make sure you have the following prerequisites installed:

- Visual Studio Code
- Remote - SSH Extension

Quick steps to install the *Remote - SSH* extension:

BEST PRACTICES FOR USING WATGPU

- Implement **checkpointing**!
  - Jobs may be **requeued** by priority jobs, nodes might go down for maintenance

- Interactive jobs are meant for specific tasks (debugging/testing)
  - **Interactive jobs can’t be requeued** (so if Slurm fails, all progress is lost)
  - Resources aren’t automatically relinquished

- Indirect way of monitoring job performance
  - nvidia-smi -a | grep Serial
  - Monitor GPU utilization level at watgpu.cs.uwaterloo/current_state.html

ADVANCED USAGE

- Using SIGTERM signals with `--signal=[R:]<sig_num>[@sig_time]`
- Shared directories `/share/<group_name>/.`
- Batch script job arrays for large number of independent jobs (parameter sweeps, simulations, processing large number of data files…)
- Multithreaded/MPI jobs

UNIVERSITY OF WATERLOO

FACULTY OF MATHEMATICS

Lucas Gamez (lmgamez@uwaterloo.ca), DC 2607
Indy Ng (wk5ng@uwaterloo.ca), DC 2607
Lori Paniak (ldpaniak@uwaterloo.ca), DC 2625