SIT Slurm Service User Guide

This guide is intended only for users who have received their account details by email and need to use Slurm through “slurm-service”.

Accessing the Service

1. Connect to “slurm-service” via SSH using the username and password provided by email on port 2222.

   Example: $ ssh *<username>*@slurm-service.sit.kmutt.ac.th -p 2222

   After logging in, you will see the following message:

   ==========

   SIT Slurm Service

   ==========

   This node is for SSH and Slurm submission only.

   Use compute resources through “srun” or “sbatch”.

   GPU is available on slurm-compute, not here.

   ==========

   <username>@slurm-service: $ _
2. Run jobs through Slurm only, using commands such as “sbatch” or “srun”.

User Workspace Structure

- Each user has a separate workspace at “/workspace/slot[*x*]”.
- Data in your slot is not shared with other slots.
- Do not store important data outside your own workspace.
- When your access expires, the system will automatically delete your account and all data in your slot.

Submitting Jobs to Slurm

1. **Create your own job file.**

Create a “myjob.sbatch” file based on the template and save it in your workspace.

Template Example

Before running a production workload, use this template as your primary reference to manage VRAM and the environment in accordance with system requirements. Jobs that do not comply with these requirements may not run successfully.

```
#!/bin/bash
#SBATCH --job-name=myjob
#SBATCH --partition=compute
#SBATCH --gres=shard:1
#SBATCH --mem=24G
#SBATCH --begin=now
#SBATCH --time=02:00:00
#SBATCH --output=job.out
#SBATCH --error=job.err

set -euo pipefail

TARGET_VRAM_GB=15.0
export TARGET_VRAM_GB
export PYTHONUNBUFFERED="${PYTHONUNBUFFERED:-1}"

python3 - <<'PY'
import os
import torch
```

```
target_vram_gb = float(os.environ["TARGET_VRAM_GB"])
device = torch.device("cuda:0")
torch.cuda.set_device(device)

total_gb = torch.cuda.get_device_properties(device).total_memory / (1024 ** 3)
fraction = min(0.99, target_vram_gb / total_gb)
torch.cuda.memory.set_per_process_memory_fraction(fraction, device=device)

# Put your workload here.
PY
```

Additional Explanation

- "--job-name" sets the name of your job.
- "--time" sets the maximum run time in "HH:MM:SS" format. The job will time out when this duration is reached.
- Add your own code after "# Put your workload here."

2. **Submit a Job**

   Command: "sbatch *<file>*"

   Example: $ sbatch myjob.sbatch

   The system will start the job and return a job ID, such as "Submitted batch job 12345".

3. **View Running or Queued Jobs**
   - View the entire queue: "squeue"
   - View a specific job: "squeue -j *<jobid>*". The command displays the status, such as "R=Running" or "P=Pending".
   - View only your own jobs: "squeue -u $USER"
   - View job details: "scontrol show job *<jobid>*"
4. **Output and Error Files**

When “#SBATCH --output=job.out” and “#SBATCH --error=job.err” are configured in the job file, output and error messages are written directly to these files.

5. **Cancel a Job**

   Command: “scancel *<jobid>”*

Usage Requirements

- The system uses a host-side VRAM killer to prevent any user from exceeding the policy limit of 15 GB of VRAM per user.
- The system calculates the combined VRAM usage of all processes belonging to the same user.
- If a user’s total VRAM usage exceeds 15 GB, all processes belonging to that user will be terminated.
- Allow user to run more than one job at the same time.
- When running multiple jobs concurrently, allocate the VRAM budget carefully so that the combined usage does not exceed 15 GB.
- The system checks VRAM usage every 5 seconds.
- If a workload exceeds the threshold, the system first sends a termination signal and then kills the process if it does not stop.
- When a process is killed, a notification file is written to the workspace of that slot.
- When a process is killed, the files in your workspace are not deleted, but your job will stop running.
- The notification file is named “vram_killer_notice” and is stored in the user workspace.
- Store data only in your own “/workspace/slot[*x*]” directory. Do not save data outside your assigned area.
- You will not see the GPU when you SSH into this server. When you submit a job, the system automatically sends it to the GPU-enabled compute node.

Recommendations:

- Start with a small job.
- Check that “torch.cuda.is_available()” works correctly.
- Check VRAM usage before and after allocation.
- Do not run multiple jobs concurrently if their combined usage may exceed 15 GB.

Dashboard

View the system overview on the dashboard at http://slurm-service.sit.kmutt.ac.th:8080

The dashboard displays the following basic information:

- Occupied and available slots
- Number of running jobs
- Host and GPU utilization
- Slot details
    - Username
    - Expiration date
    - User utilization
    - Names of running jobs
    - Number of running jobs
- The dashboard refreshes automatically every 10 seconds and includes a Refresh button for manual updates.

Expiration Date

- Your expiration date is provided by email together with your account details.

- When your access is close to expiring and you need an extension, contact your instructor and the responsible staff.
- After your account expires, the system automatically deletes the data in your assigned slot.
- Back up your data outside the system before your expiration date if you need to keep it.