How to use a supercomputer

SLURM Job Scheduler

Speaker: Martin Pfister
HPC / AI Team, EuroCC Austria

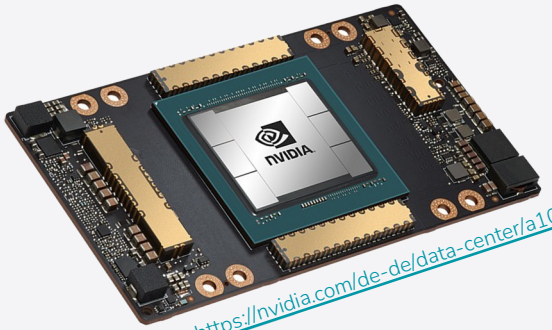
EURO AUSTRIA

The Vienna Scientific Cluster

VSC-4 (2019)

790 CPU nodes

- 2x Intel Skylake Platinum CPUs
- 2x 24 cores per CPU
- 96 GB of memory per node



source: https://nvidia.com/de-de/data-center/a100/

VSC-5 (2022)

770 CPU nodes

- 2x AMD EPYC Milan
- 2x 64 cores per CPU
- 512 GB of memory per node

60 GPU nodes 2x NVIDIA A100,

- 40 GB memory per GPU

40 GPU nodes 2x NVIDIA A40

- 48 GB memory per GPU

EURO AUSTRIA

Need More Compute Power?

LUMI

- #5 in Top500
- Linpack: 380 PFlop/s
- AMD EPYC CPUs
- AMD Instinct MI250X GPUs (128 GB)

https://www.lumi-supercomputer.eu/

Leonardo

- #7 in Top500
- Linpack: 240 Pflos/p
- Intel Xeon CPUs
- NVIDIA A100 GPUs (64GB)

https://leonardo-supercomputer.cineca.eu/

EURO AUSTRIA

Supercomputers in Europe

EuroHPC JU systems

Apply for access to EuroHPC supercomputers

Different access modes:

- Benchmark Access
- Development Access
- ...
- Extreme Scale Access

https://eurohpc-ju.europa.eu/

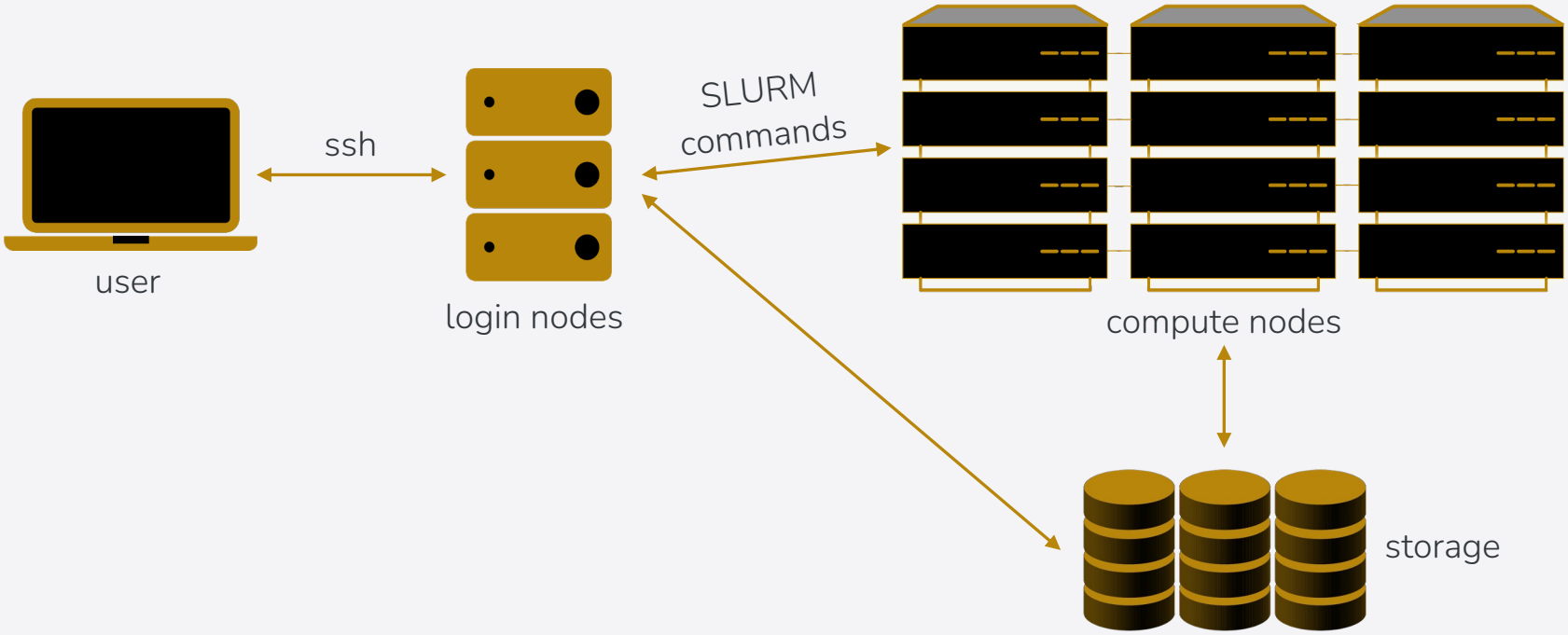
EURO AUSTRIA

Supercomputers in Europe

- VSC5 (60 nodes with 2 Nvidia A100 40 GB)
- LEONARDO (3456 nodes with 4 Nvidia A100 64 GB)
- LUMI-G (2978 nodes with 4 AMD MI250x 128 GB)
- MUSICA (~200 nodes with 4 Nvidia H100 96 GB)

https://eurohpc-ju.europa.eu/supercomputers/our-supercomputers_en

Typical Setup of a Supercomputer



SLURM: Job Scheduler

VSC5 job script (1 GPU)

```
#!/bin/bash
#SBATCH --partition=zen3_0512_a100x2
#SBATCH --qos=zen3_0512_a100x2


#SBATCH --gres=gpu:1              # Number of GPUs (1 or 2 on VSC5)


#SBATCH --time=3-00:00:00         # Time limit. Format: Days-hours:minutes:seconds

<whatever command should be executed on the compute node>
```

SLURM: Job Scheduler

VSC5 job script (1 GPU)

```
#!/bin/bash
#SBATCH --partition=zen3_0512_a100x2
#SBATCH --qos=zen3_0512_a100x2


#SBATCH --gres=gpu:1              # Number of GPUs (1 or 2 on VSC5)


#SBATCH --time=3-00:00:00         # Time limit. Format: Days-hours:minutes:seconds

module purge                      # Start in a clean environment
module load miniconda3            # Load conda

conda run -n conda_env_name --no-capture-output python -c "import torch;
print(torch.__version__); print(torch.cuda.get_device_properties(0))"
```

EURO AUSTRIA

SLURM: Job Scheduler

VSC5 job script (1 GPU)

```
#!/bin/bash
#SBATCH --partition=zen3_0512_a100x2
#SBATCH --qos=zen3_0512_a100x2


#SBATCH --gres=gpu:1              # Number of GPUs (1 or 2 on VSC5)


#SBATCH --time=3-00:00:00         # Time limit. Format: Days-hours:minutes:seconds

module purge                      # Start in a clean environment
module load miniconda3            # Load conda

conda run -n conda_env_name --no-capture-output python script.py
```

SLURM: Job Scheduler

VSC5 job script (2 GPUs)

```
#!/bin/bash
#SBATCH --partition=zen3_0512_a100x2
#SBATCH --qos=zen3_0512_a100x2
#SBATCH --nodes=1                # Number of nodes
#SBATCH --ntasks-per-node=1      # Number of `srun` tasks executed per node
#SBATCH --gres=gpu:2             # Number of GPUs (1 or 2 on VSC5)


#SBATCH --time=3-00:00:00        # Time limit. Format: Days-hours:minutes:seconds

module purge                     # Start in a clean environment
module load miniconda3           # Load conda

conda run -n conda_env_name --no-capture-output python script.py
```

SLURM: Job Scheduler

VSC5 job script (2 nodes)

```
#!/bin/bash
#SBATCH --partition=zen3_0512_a100x2
#SBATCH --qos=zen3_0512_a100x2
#SBATCH --nodes=2                # Number of nodes
#SBATCH --ntasks-per-node=1      # Number of `srun` tasks executed per node
#SBATCH --gres=gpu:2             # Number of GPUs (1 or 2 on VSC5)


#SBATCH --time=3-00:00:00        # Time limit. Format: Days-hours:minutes:seconds

module purge                     # Start in a clean environment
module load miniconda3           # Load conda

srun bash -c "conda run -n conda_env_name --no-capture-output python script.py"
```

SLURM: Job Scheduler

Leonardo job script (2 nodes)

```
#!/bin/bash
#SBATCH --partition=boost_usr_prod
#SBATCH --qos=boost_qos_dbg      # High priority QOS. Remove line for normal priority.
#SBATCH --nodes=2                # Number of nodes
#SBATCH --ntasks-per-node=1      # Number of `srun` tasks executed per node
#SBATCH --gpus-per-task=4        # Number of GPUs (up to 4 on Leonardo)
#SBATCH --mem-per-gpu=120GB
#SBATCH --cpus-per-task=32       # should be 8 * gpus-per-task on Leonardo
#SBATCH --time=0:30:00           # up to 0:30:00 for boost_qos_dbg

module purge                     # Start in a clean environment
module load anaconda3            # Load conda

srun bash -c "conda run -n conda_env_name --no-capture-output python script.py"
```

SLURM: Job Scheduler

Useful SLURM commands

```
# Submit a job:
$ sbatch job.sh

# Check submitted jobs:
$ squeue --me

# Watch the output as the job runs:
$ tail -c +0 -f slurm-<job_id>.out

# Cancel job:
$ scancel <job_id>

# Get a shell at a node while a job is running:
$ ssh <compute_node_name>
# or:
$ srun --overlap --pty -jobid=<job_id> bash
```

STAY IN TOUCH

EuroCC Austria

@eurocc_austria

eurocc-austria.at

EURO AUSTRIA

THANK YOU

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101101903. The JU receives support from the Digital Europe Programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Türkiye, Republic of North Macedonia, Iceland, Montenegro, Serbia