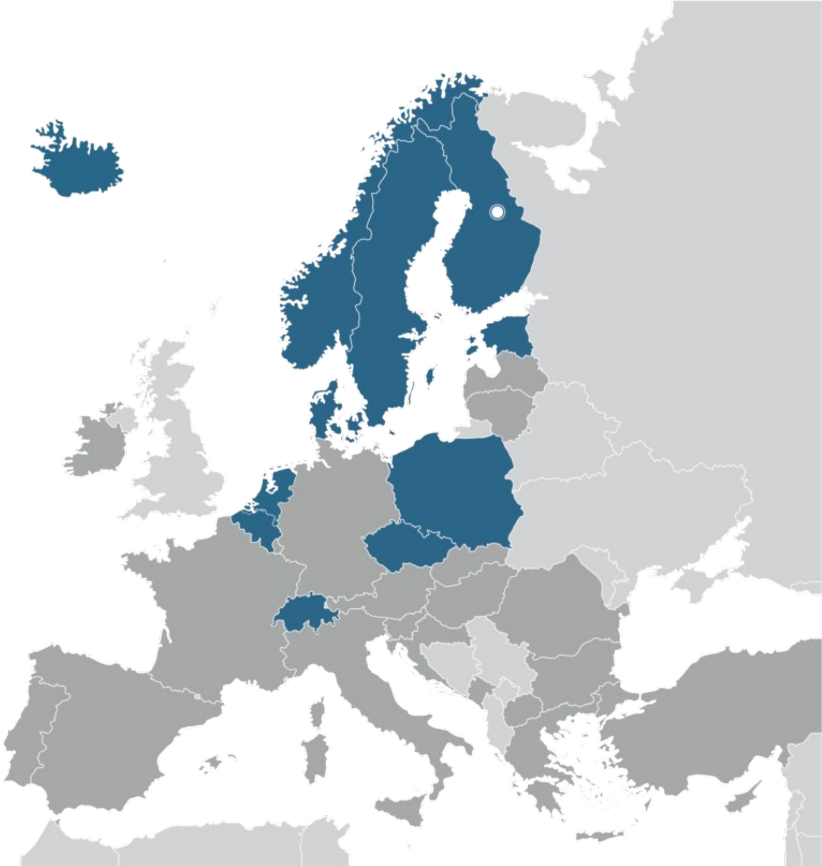
PyTorch on LUMI

e-INFRA CZ conference, 29.-30.04.2024
Lukas Prediger (CSC – IT Center for Science)

CSC and LUMI

- LUMI consortium countries: Finland, Belgium, the Czech Republic, Denmark, Estonia, Iceland, the Netherlands, Norway, Poland, Sweden, and Switzerland
- Hosted by CSC in Kajaani, Finland
- Main support channel: LUMI User Support
- Science support from local organizations like CSC
- Resource allocation:
  - 50% EuroHPC JU
  - **each consortium country** according to contribution
  - up to 20% industry and SMEs



Goal for today: Running PyTorch on LUMI and avoiding common pitfalls

LUMI

- Example: Finetuning GPT-Neo LLM for generating movie reviews on the IMDb data set
- Using huggingface.co's 🤗 datasets and transformers on top of `PyTorch` as training library…
- … but the exact training code / library does not matter that much

GPT-Neo 1.3B

- 1370 M parameters
- BF16
- ~2.67 GB

Stanford IMDb data set

- 100 000 movie reviews
- Varying lengths (low hundreds of words)
- 25 000 reserved for testing

All example scripts available @

https://github.com/CSCfi/eINFRACZ-2024-04-PyTorch-on-LUMI

How LUMI is different

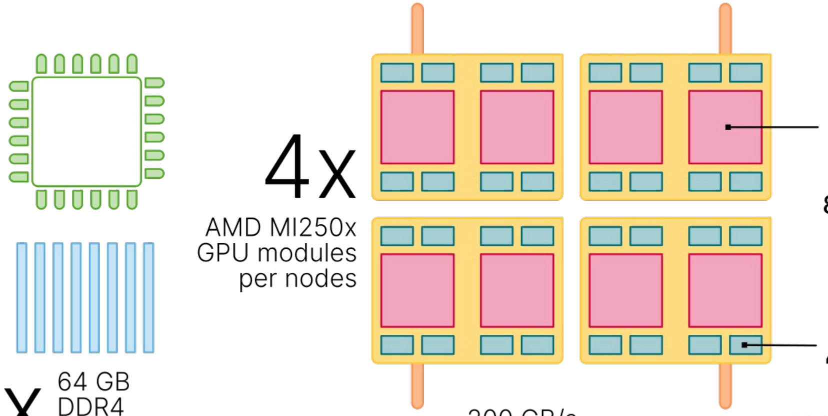
LUMI

- Massive system shared with many other users
- SLURM for resource allocation and scheduling of jobs
- AMD instead of Nvidia GPUs: ROCm, not CUDA
- Only network storage – no local storage

LUMI-G (GPU-partition) high level description

2978x compute nodes

23824x GCD total



1x 64 cores AMD EPYC 7A53

8x 64 GB DDR4 memory

512 GB total

4x AMD MI250x GPU modules per nodes

2x Graphics Compute Die (GCD) per module

8x per compute node

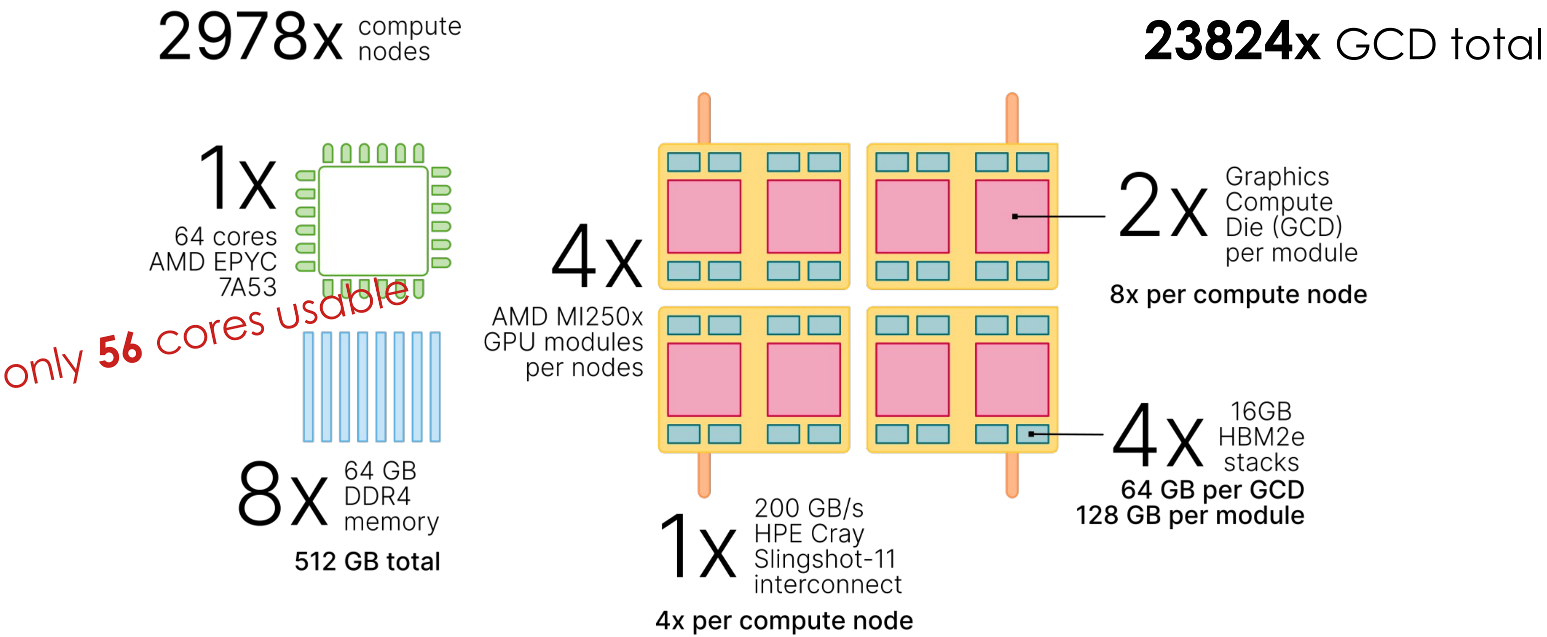
4x 16GB HBM2e stacks

64 GB per GCD
128 GB per module

1x 200 GB/s HPE Cray Slingshot-11 interconnect

4x per compute node

LUMI-G (GPU-partition) high level description



2978x compute nodes

1x 64 cores AMD EPYC 7A53

only **56** cores usable

8x 64 GB DDR4 memory

512 GB total

4x AMD MI250x GPU modules per nodes

1x 200 GB/s HPE Cray Slingshot-11 interconnect

4x per compute node

23824x GCD total

2x Graphics Compute Die (GCD) per module

8x per compute node

4x 16GB HBM2e stacks

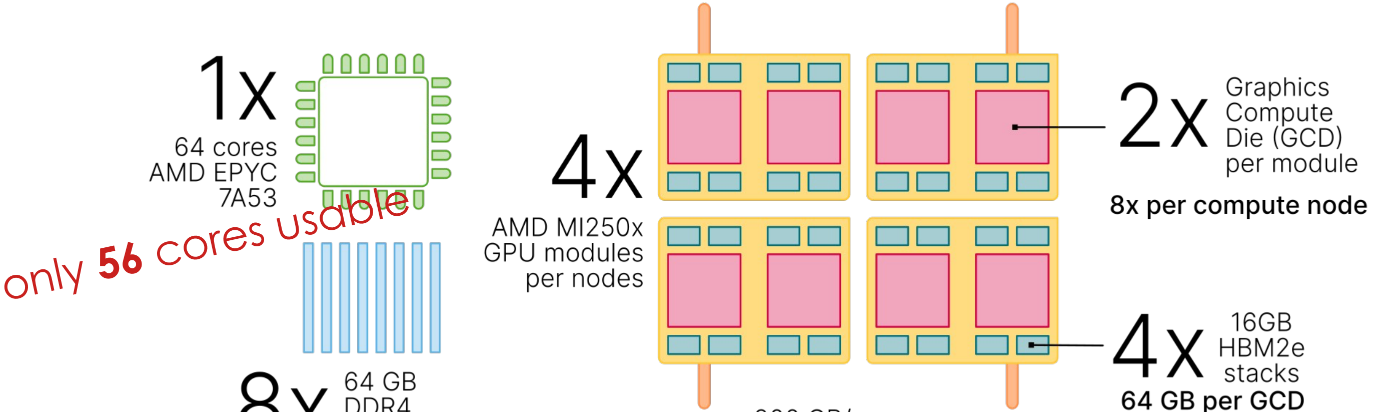
64 GB per GCD

128 GB per module

LUMI-G (GPU-partition) high level description

2978x compute nodes

23824x GCD total



1x 64 cores AMD EPYC 7A53

only **56** cores usable

8x 64 GB DDR4 memory

512 GB total

4x AMD MI250x GPU modules per nodes

2x Graphics Compute Die (GCD) per module

8x per compute node

4x 16GB HBM2e stacks

64 GB per GCD

128 GB per module

1x 200 GB/s HPE Cray Slingshot-11 interconnect

4x per compute node

Usable per GCD: 7 CPU cores, (up to) 64 GB RAM

ROCm is not CUDA

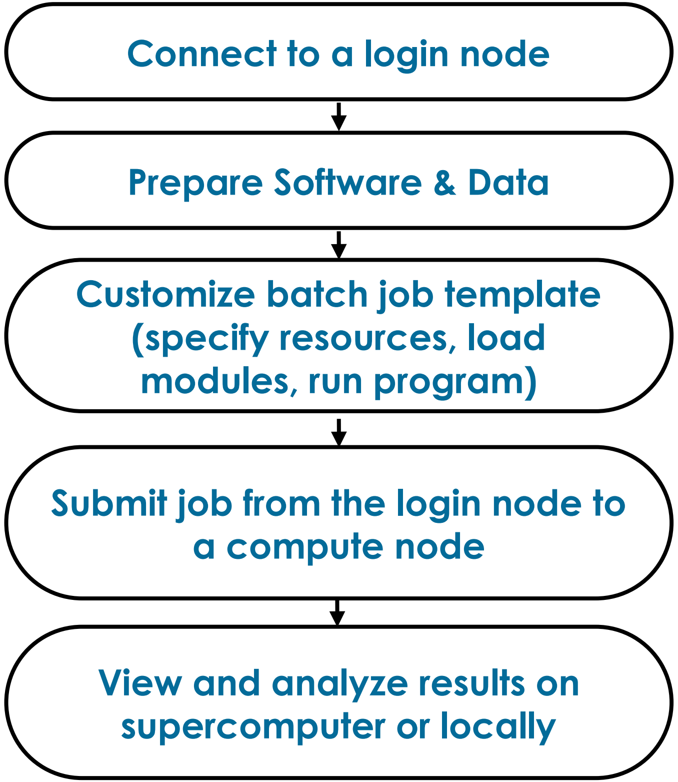
- PyTorch pretty well supported out-of-the-box
- AMD has ported and optimized many popular ML specific libraries:
  - Flash Attention, bitsandbytes, vLLM, …
- !! Beware !! Some bugs / rough edges persist
  → Usually fixable by tweaking configuration via environment variables
- !! Beware !! ROCm driver version on LUMI currently outdated
  → Upgrade coming soon!

AMD
@AMD

.@PyTorch founder @soumithchintala on AMD Day 0 support: Developers can train and deploy #AI models compatible "out of the box" with @AMDInstinct accelerators. "Simply put, when a new model is released, it's up and running on AMD platforms immediately for developers."

What does using a supercomputer look like?

LUMI



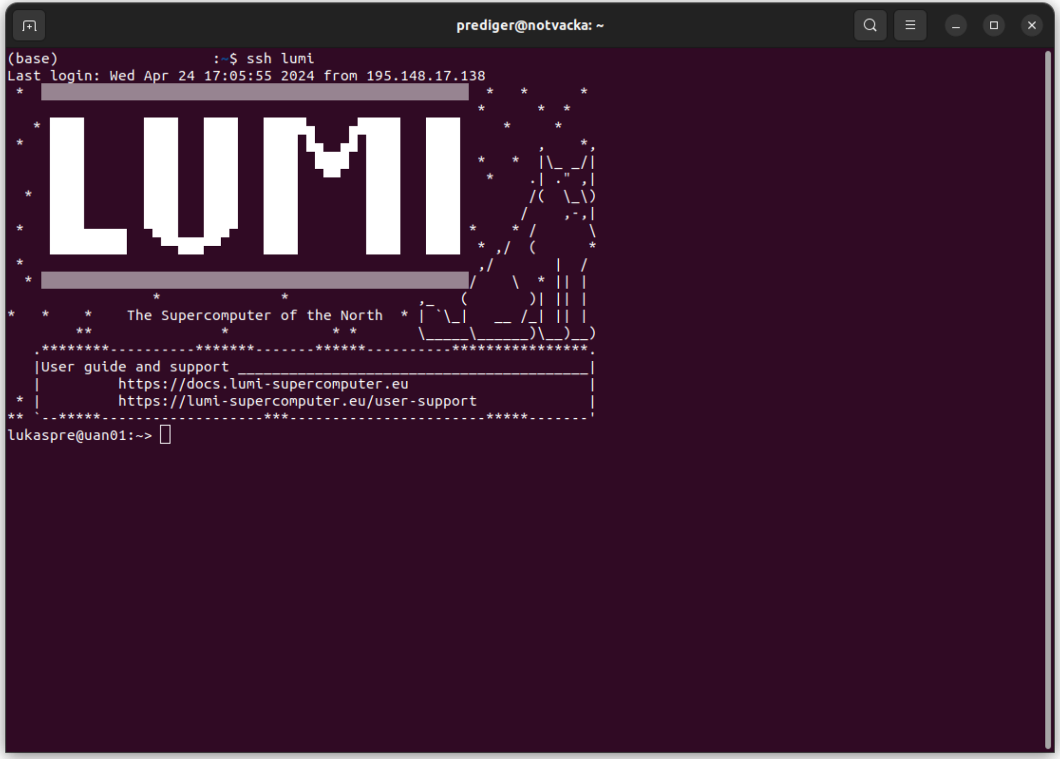
Connect to a login node

Prepare Software & Data

Customize batch job template (specify resources, load modules, run program)

Submit job from the login node to a compute node

View and analyze results on supercomputer or locally



```
(base)                   :~$ ssh lumi
Last login: Wed Apr 24 17:05:55 2024 from 195.148.17.138

*   *   *    The Supercomputer of the North  *

 .********---------*******------******---------****************.
 |User guide and support ______________________________________|
 |         https://docs.lumi-supercomputer.eu                  |
* |         https://lumi-supercomputer.eu/user-support          |
** `--*****-------------------***--------------------*****------'
lukaspre@uan01:~> 
```

Network Storage

LUMI

├ `/users/username/`

- ↳ User home (personal configuration, etc)
- ↳ 20 GB Quota

├ `/project/project_123456/`

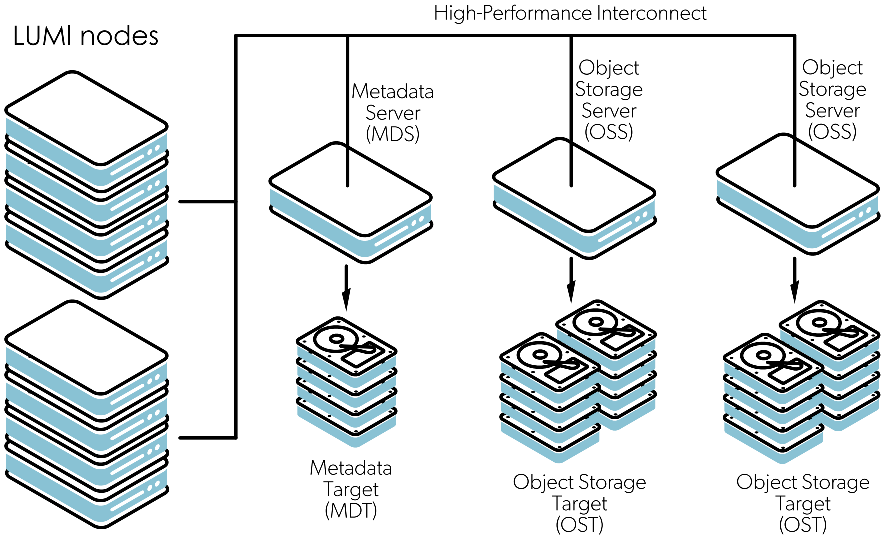
- ↳ Project home and shared files (Software)
- ↳ 50 GB Quota

├ `/scratch/project_123456/`

- ↳ Temporary storage (Checkpoints)
- ↳ 240 GB/s total
- ↳ 50 TB Quota

├ `/flash/project_123456/`

- ↳ High speed temporary storage
- ↳ 1740 GB/s total
- ↳ 2 TB Quota


High-Performance Interconnect
LUMI nodes
Metadata Server (MDS)
Object Storage Server (OSS)
Object Storage Server (OSS)
Metadata Target (MDT)
Object Storage Target (OST)
Object Storage Target (OST)

https://docs.lumi-supercomputer.eu/storage/

Network Storage

LUMI

├ `/users/username/`

  ↳ User home (personal configuration, etc)

  ↳ 20 GB Quota

├ `/project/project_123456/`

  ↳ Project home and shared files (Software)

  ↳ 50 GB Quota

├ `/scratch/project_123456/`

  ↳ Temporary storage (Checkpoints)

  ↳ 240 GB/s total

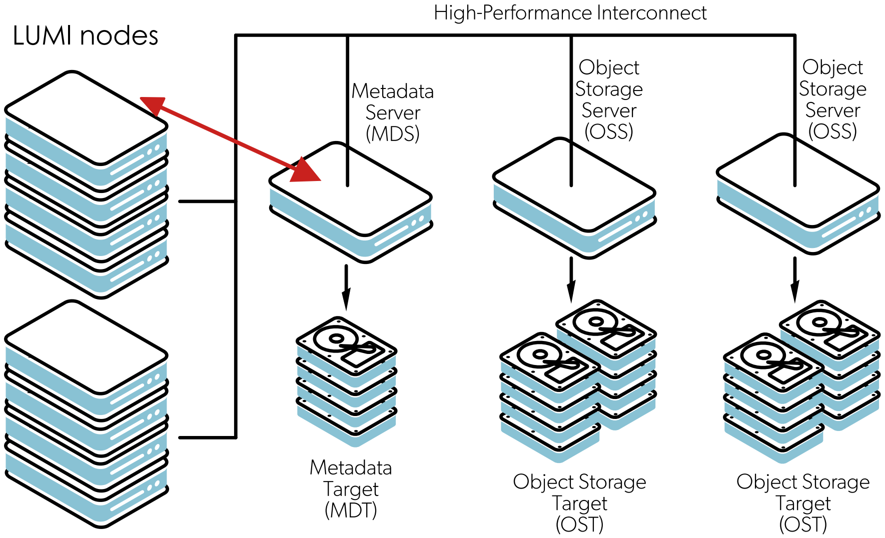
  ↳ 50 TB Quota

├ `/flash/project_123456/`

  ↳ High speed temporary storage

  ↳ 1740 GB/s total

  ↳ 2 TB Quota


High-Performance Interconnect
LUMI nodes
Metadata Server (MDS)
Object Storage Server (OSS)
Object Storage Server (OSS)
Metadata Target (MDT)
Object Storage Target (OST)
Object Storage Target (OST)

https://docs.lumi-supercomputer.eu/storage/

Network Storage

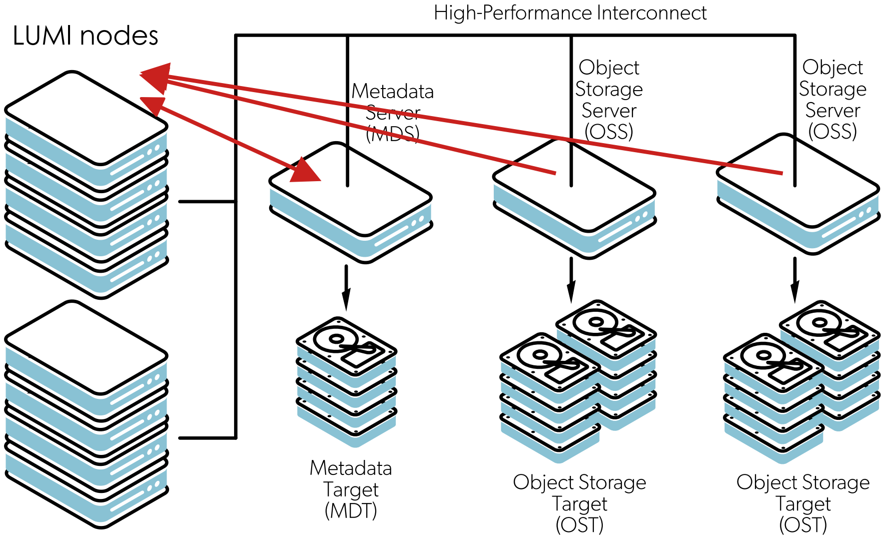
LUMI

├ /users/username/
- ↳ User home (personal configuration, etc)
- ↳ 20 GB Quota

├ /project/project_123456/
- ↳ Project home and shared files (Software)
- ↳ 50 GB Quota

├ /scratch/project_123456/
- ↳ Temporary storage (Checkpoints)
- ↳ 240 GB/s total
- ↳ 50 TB Quota

├ /flash/project_123456/
- ↳ High speed temporary storage
- ↳ 1740 GB/s total
- ↳ 2 TB Quota


High-Performance Interconnect
LUMI nodes
Metadata Server (MDS)
Object Storage Server (OSS)
Object Storage Server (OSS)
Metadata Target (MDT)
Object Storage Target (OST)
Object Storage Target (OST)

https://docs.lumi-supercomputer.eu/storage/

Network Storage

LUMI

├ /users/username/

↳ User home (personal configuration, etc)

↳ 20 GB Quota

├ /project/project_123456/

↳ Project home and shared files (Software)

↳ 50 GB Quota

├ /scratch/project_123456/

↳ Temporary storage (Checkpoints)

↳ 240 GB/s total

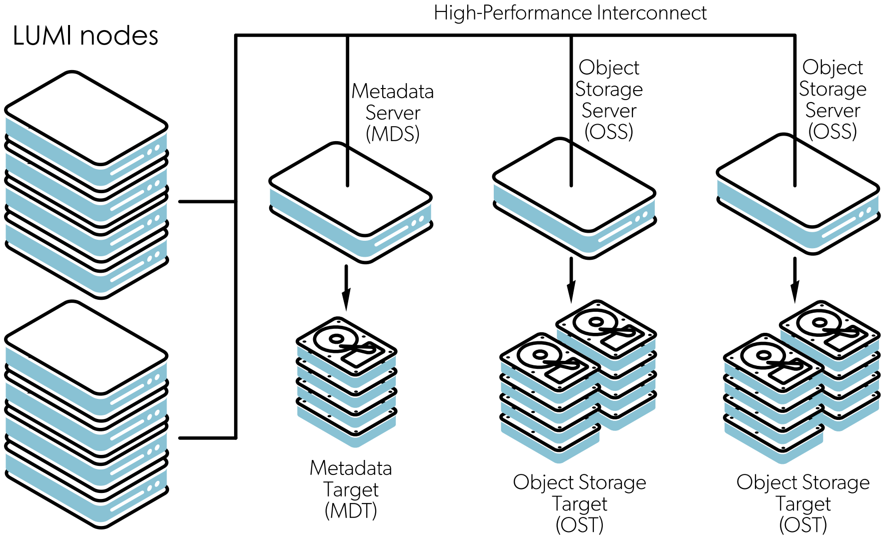
↳ 50 TB Quota

├ /flash/project_123456/

↳ High speed temporary storage

↳ 1740 GB/s total

↳ 2 TB Quota


High-Performance Interconnect
LUMI nodes
Metadata Server (MDS)
Object Storage Server (OSS)
Object Storage Server (OSS)
Metadata Target (MDT)
Object Storage Target (OST)
Object Storage Target (OST)

https://docs.lumi-supercomputer.eu/storage/

Network Storage

LUMI

├ `/users/username/`

- ↳ User home (personal configuration, etc)
- ↳ 20 GB Quota

├ `/project/project_123456/`

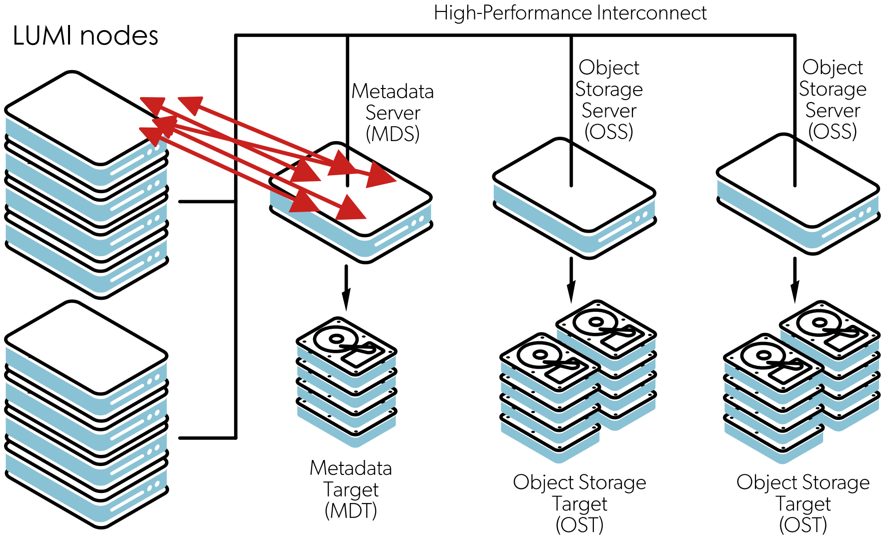
- ↳ Project home and shared files (Software)
- ↳ 50 GB Quota

├ `/scratch/project_123456/`

- ↳ Temporary storage (Checkpoints)
- ↳ 240 GB/s total
- ↳ 50 TB Quota

├ `/flash/project_123456/`

- ↳ High speed temporary storage
- ↳ 1740 GB/s total
- ↳ 2 TB Quota


High-Performance Interconnect
LUMI nodes
Metadata Server (MDS)
Object Storage Server (OSS)
Object Storage Server (OSS)
Metadata Target (MDT)
Object Storage Target (OST)
Object Storage Target (OST)

https://docs.lumi-supercomputer.eu/storage/

Network Storage

LUMI

├ /users/username/

↳ User home (personal configuration, etc)

↳ 20 GB Quota

├ /project/project_123456/

↳ Project home and shared files (Software)

↳ 50 GB Quota

├ /scratch/project_123456/

↳ Temporary storage (Checkpoints)

↳ 240 GB/s total

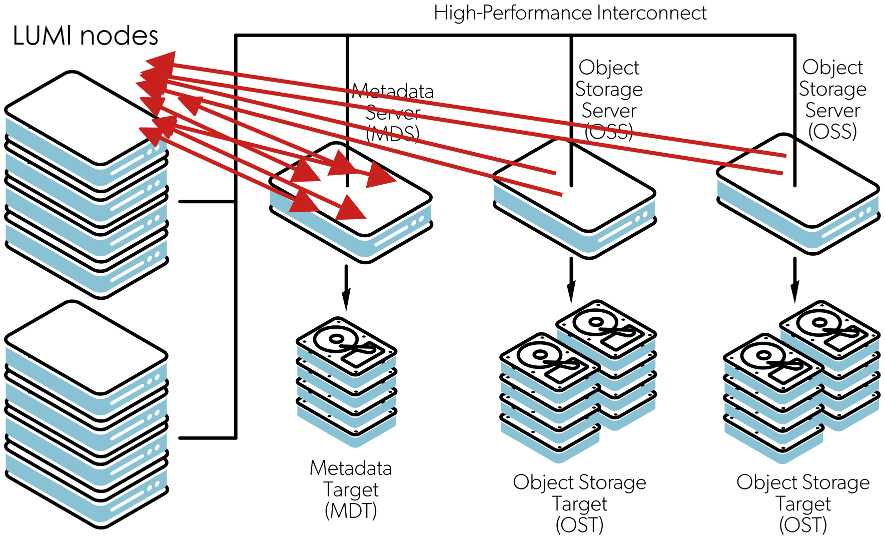
↳ 50 TB Quota

├ /flash/project_123456/

↳ High speed temporary storage

↳ 1740 GB/s total

↳ 2 TB Quota


High-Performance Interconnect
LUMI nodes
Metadata Server (MDS)
Object Storage Server (OSS)
Object Storage Server (OSS)
Metadata Target (MDT)
Object Storage Target (OST)
Object Storage Target (OST)

https://docs.lumi-supercomputer.eu/storage/

Network Storage

├ `/users/username/`

- ↳ User home (personal configuration, etc)
- ↳ 20 GB Quota

├ `/project/project_123456/`

- ↳ Project home and shared files (Software)
- ↳ 50 GB Quota

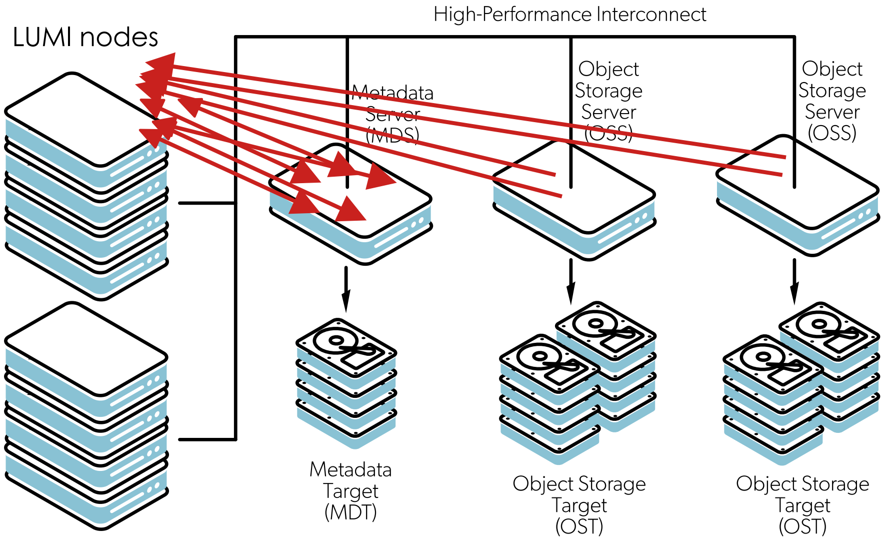
├ `/scratch/project_123456/`

- ↳ Temporary storage (Checkpoints)
- ↳ 240 GB/s total
- ↳ 50 TB Quota

├ `/flash/project_123456/`

- ↳ High speed temporary storage
- ↳ 1740 GB/s total
- ↳ 2 TB Quota

Avoid creating/accessing large amounts of small files

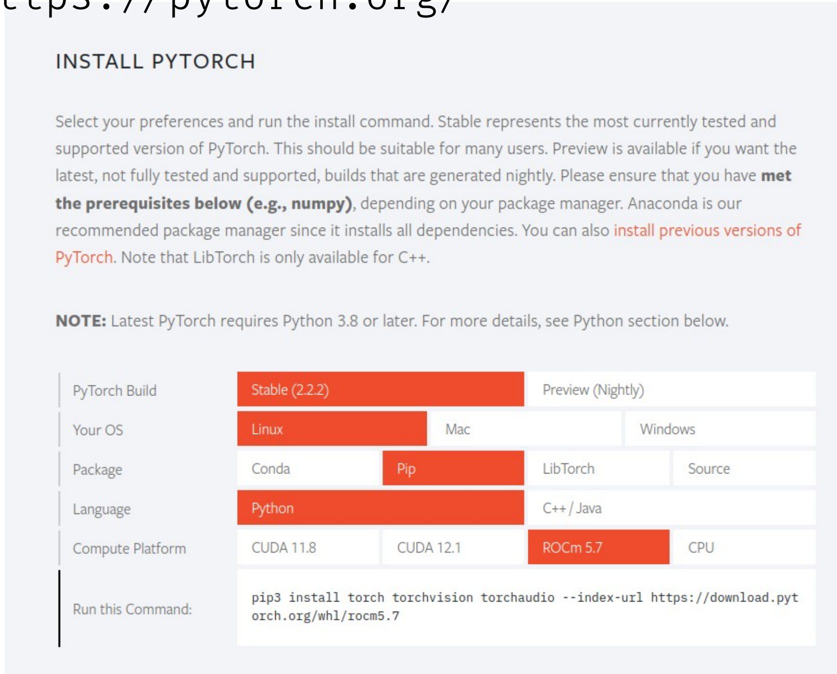


High-Performance Interconnect
LUMI nodes
Metadata Server (MDS)
Object Storage Server (OSS)
Object Storage Server (OSS)
Metadata Target (MDT)
Object Storage Target (OST)
Object Storage Target (OST)

https://docs.lumi-supercomputer.eu/storage/

Setting up the Software Environment

https://pytorch.org/



INSTALL PYTORCH

Select your preferences and run the install command. Stable represents the most currently tested and supported version of PyTorch. This should be suitable for many users. Preview is available if you want the latest, not fully tested and supported, builds that are generated nightly. Please ensure that you have **met the prerequisites below (e.g., numpy)**, depending on your package manager. Anaconda is our recommended package manager since it installs all dependencies. You can also install previous versions of PyTorch. Note that LibTorch is only available for C++.

NOTE: Latest PyTorch requires Python 3.8 or later. For more details, see Python section below.

PyTorch Build	[✓ Stable (2.2.2)]	Preview (Nightly)		
Your OS	[✓ Linux]	Mac	Windows	
Package	Conda	[✓ Pip]	LibTorch	Source
Language	[✓ Python]	C++ / Java		
Compute Platform	CUDA 11.8	CUDA 12.1	[✓ ROCm 5.7]	CPU

Run this Command:

```
pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/rocm5.7
```

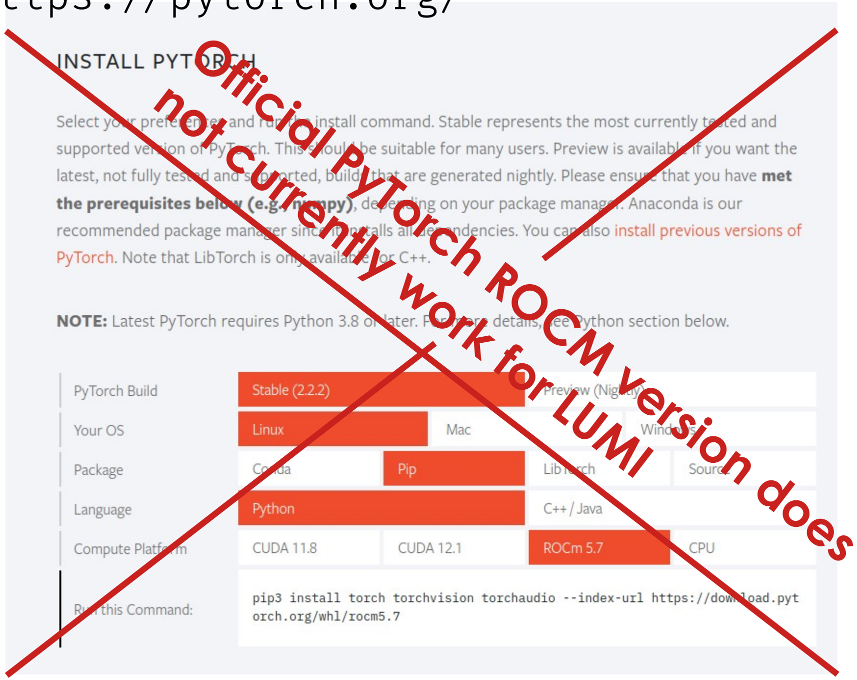
Container icon: Iconjam @ flaticon.com

https://docs.lumi-supercomputer.eu/software/

Setting up the Software Environment

LUMI

https://pytorch.org/



INSTALL PYTORCH

Select your preferences and run the install command. Stable represents the most currently tested and supported version of PyTorch. This should be suitable for many users. Preview is available if you want the latest, not fully tested and supported, builds that are generated nightly. Please ensure that you have **met the prerequisites below (e.g., numpy)**, depending on your package manager. Anaconda is our recommended package manager since it installs all dependencies. You can also install previous versions of PyTorch. Note that LibTorch is only available for C++.

NOTE: Latest PyTorch requires Python 3.8 or later. For more details, see Python section below.

PyTorch Build	[✓ Stable (2.2.2)]	Preview (Nightly)		
Your OS	[✓ Linux]	Mac	Windows	
Package	Conda	[✓ Pip]	LibTorch	Source
Language	[✓ Python]	C++ / Java		
Compute Platform	CUDA 11.8	CUDA 12.1	[✓ ROCm 5.7]	CPU

Run this Command:

```
pip3 install torch torchvision torchaudio --index-url https://download.pyt
orch.org/whl/rocm5.7
```

Official PyTorch ROCM version does not currently work for LUMI

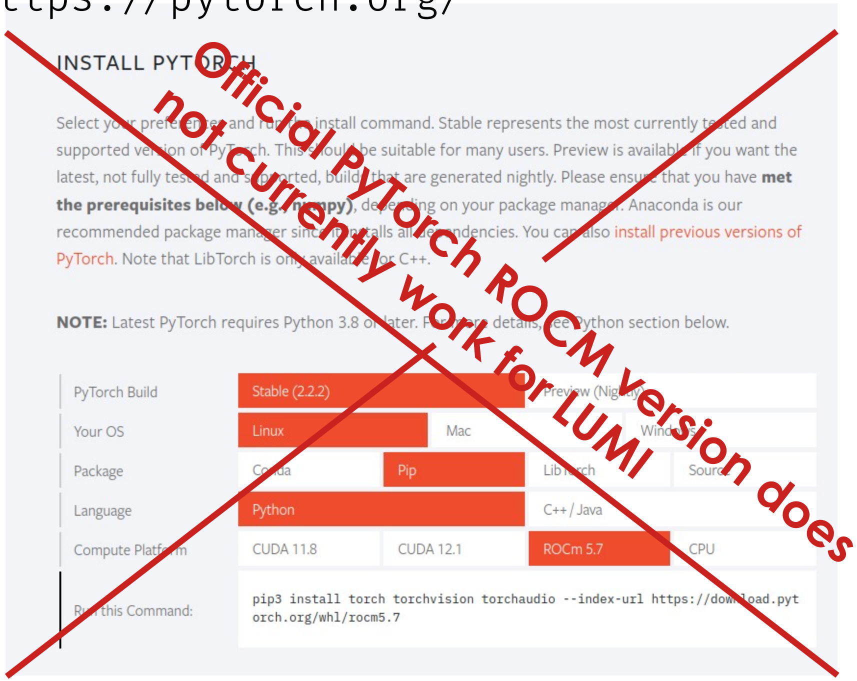
Container icon: Iconjam @ flaticon.com

https://docs.lumi-supercomputer.eu/software/

Setting up the Software Environment

LUMI

https://pytorch.org/



INSTALL PYTORCH

Select your preferences and run the install command. Stable represents the most currently tested and supported version of PyTorch. This should be suitable for many users. Preview is available if you want the latest, not fully tested and supported, builds that are generated nightly. Please ensure that you have **met the prerequisites below (e.g., numpy)**, depending on your package manager. Anaconda is our recommended package manager since it installs all dependencies. You can also install previous versions of PyTorch. Note that LibTorch is only available for C++.

NOTE: Latest PyTorch requires Python 3.8 or later. For more details, see Python section below.

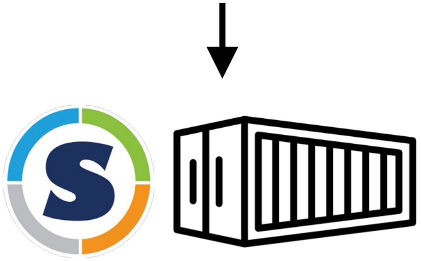
PyTorch Build	Stable (2.2.2)		Preview (Nightly)	
Your OS	Linux	Mac		Windows
Package	Conda	Pip	LibTorch	Source
Language	Python		C++ / Java	
Compute Platform	CUDA 11.8	CUDA 12.1	ROCm 5.7	CPU
Run this Command:	pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/rocm5.7			

Official PyTorch ROCM version does not currently work for LUMI

! environment-minimal.yml

```yaml
channels:
  - pytorch
  - conda-forge
  - defaults
dependencies:
  - accelerate=0.29.1
  - datasets=2.18.0
  - python=3.10.14
  - pytorch=2.2.2
  - transformers=4.39.3

```

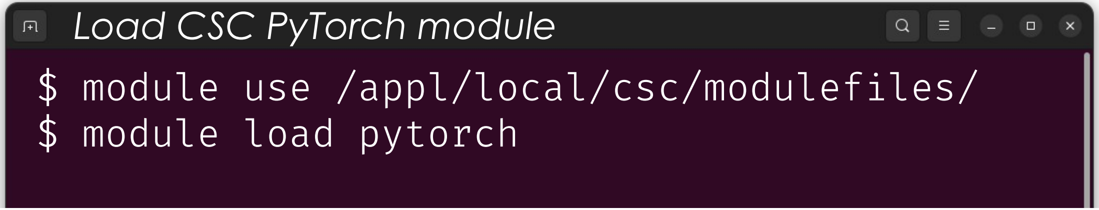
~47k files



Container icon: Iconjam @ flaticon.com

https://docs.lumi-supercomputer.eu/software/

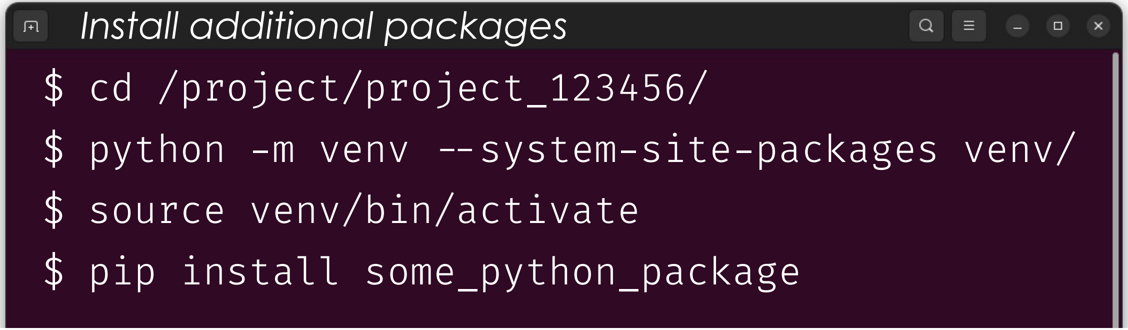
Setting up the Software Environment



Load CSC PyTorch module

```
$ module use /appl/local/csc/modulefiles/
$ module load pytorch
```

→ PyTorch + many common ML libraries
+ wrappers for the container



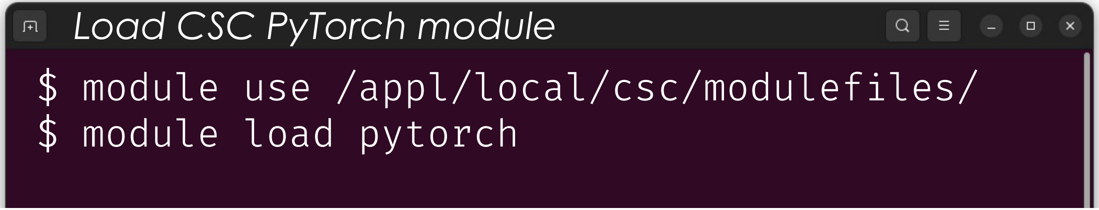
Install additional packages

```
$ cd /project/project_123456/
$ python -m venv --system-site-packages venv/
$ source venv/bin/activate
$ pip install some_python_package
```

https://docs.lumi-supercomputer.eu/software/
https://docs.lumi-supercomputer.eu/runjobs/lumi_env/Lmod_modules/

Container icon: Iconjam @ flaticon.com

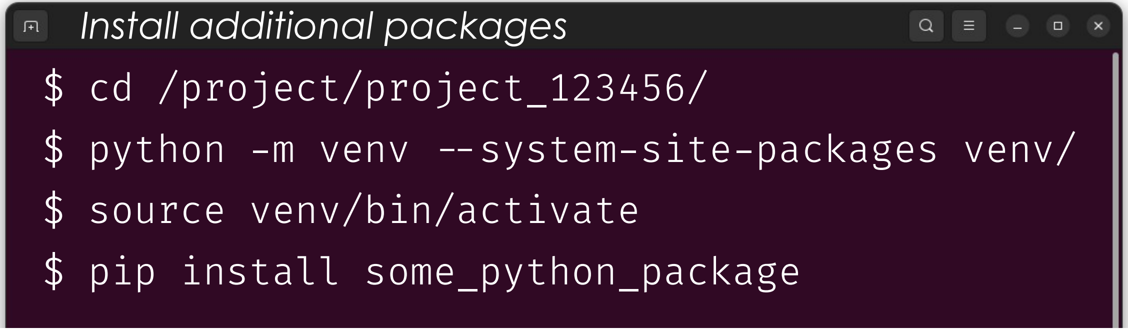
Setting up the Software Environment



Load CSC PyTorch module

```
$ module use /appl/local/csc/modulefiles/
$ module load pytorch
```

→ PyTorch + many common ML libraries
+ wrappers for the container



Install additional packages

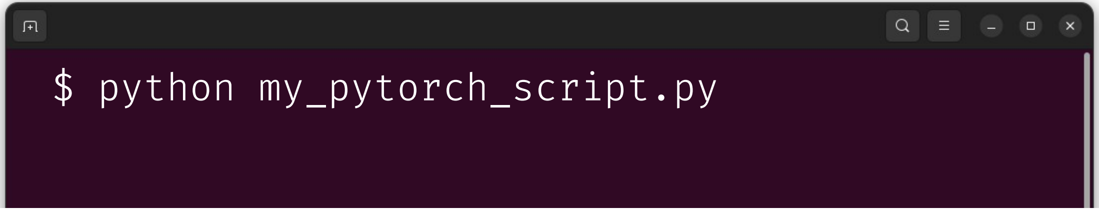
```
$ cd /project/project_123456/
$ python -m venv --system-site-packages venv/
$ source venv/bin/activate
$ pip install some_python_package
```

Only for a small number of additional packages!
If you need large custom installs
→ build your own container!

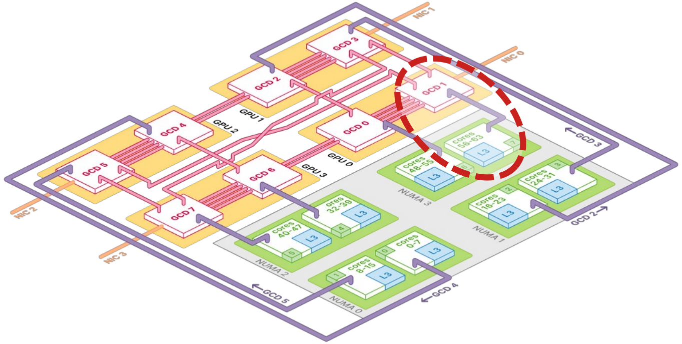
https://docs.lumi-supercomputer.eu/software/
https://docs.lumi-supercomputer.eu/runjobs/lumi_env/Lmod_modules/

Container icon: Iconjam @ flaticon.com

Training on a single Graphics Compute Die



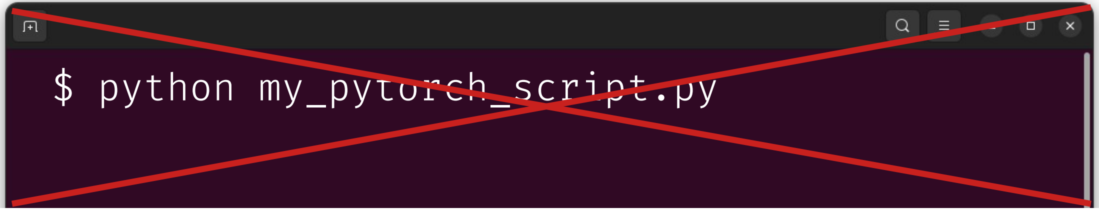
```
$ python my_pytorch_script.py
```



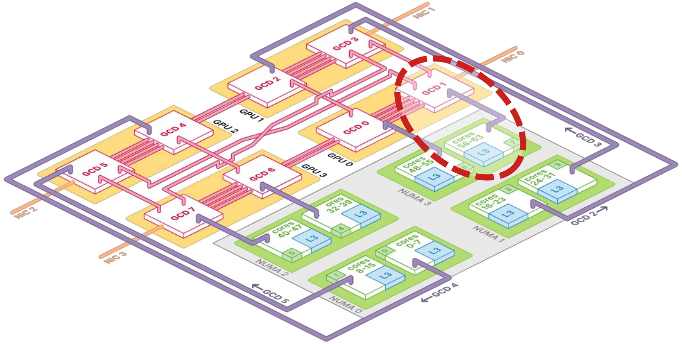
Icons: vecteezy.com

https://docs.lumi-supercomputer.eu/runjobs/

Training on a single Graphics Compute Die



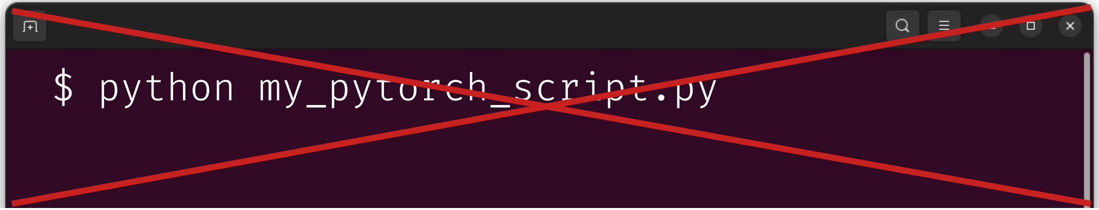
~~`$ python my_pytorch_script.py`~~



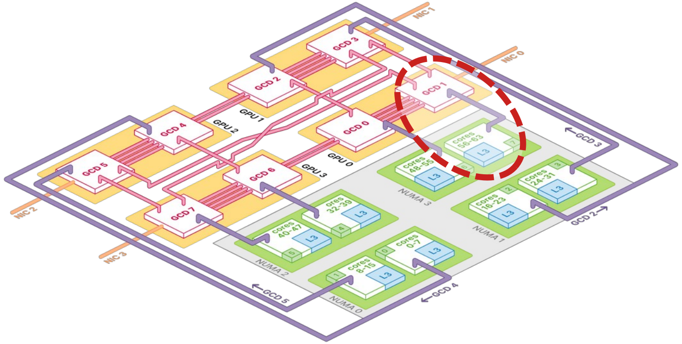
Icons: vecteezy.com

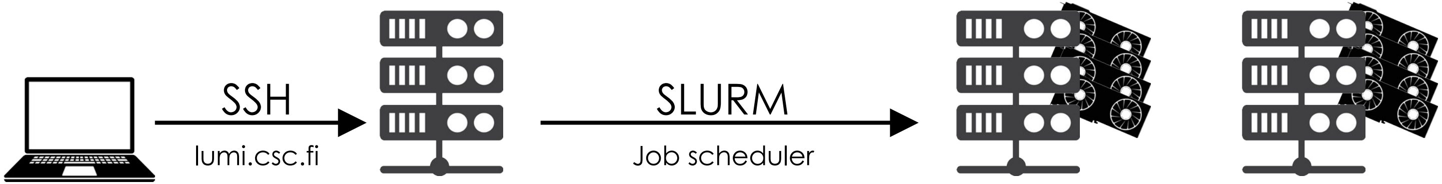
https://docs.lumi-supercomputer.eu/runjobs/

Training on a single Graphics Compute Die



~~`$ python my_pytorch_script.py`~~





SSH

lumi.csc.fi

LUMI
login node

SLURM

Job scheduler

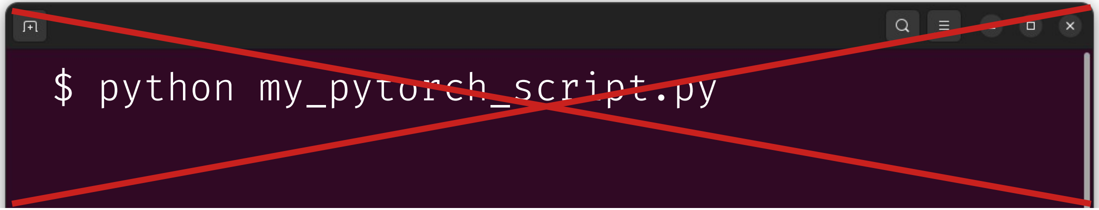
LUMI-G
compute nodes

Icons: vecteezy.com

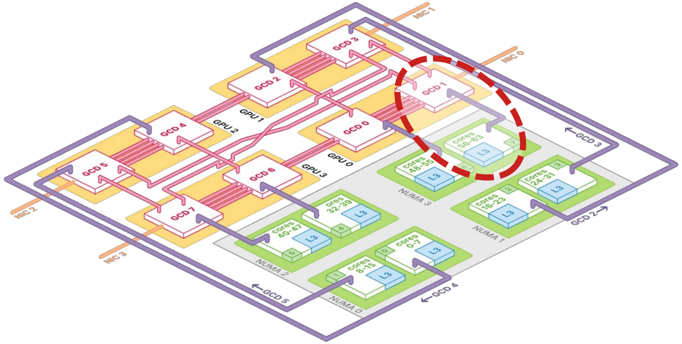
https://docs.lumi-supercomputer.eu/runjobs/

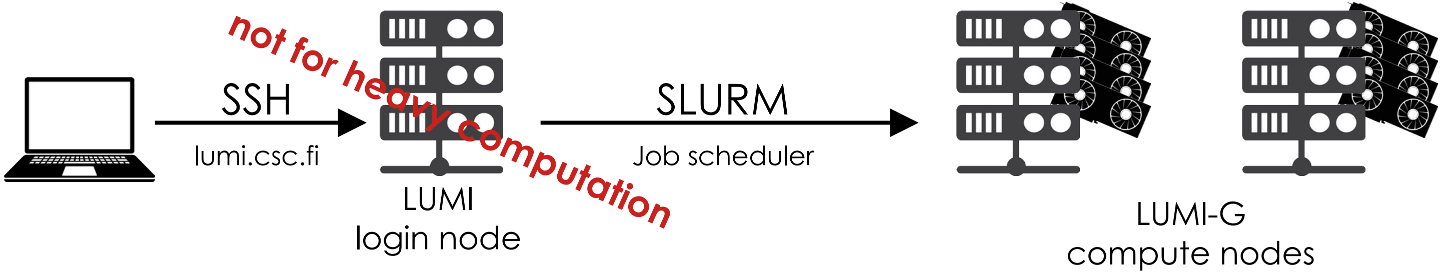
Training on a single Graphics Compute Die

LUMI



```
$ python my_pytorch_script.py
```





not for heavy computation

SSH

lumi.csc.fi

LUMI
login node

SLURM

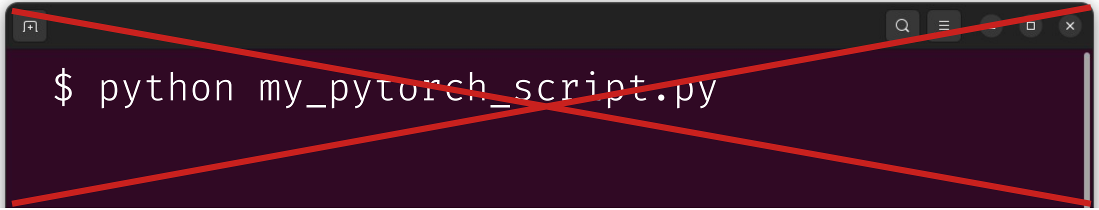
Job scheduler

LUMI-G
compute nodes

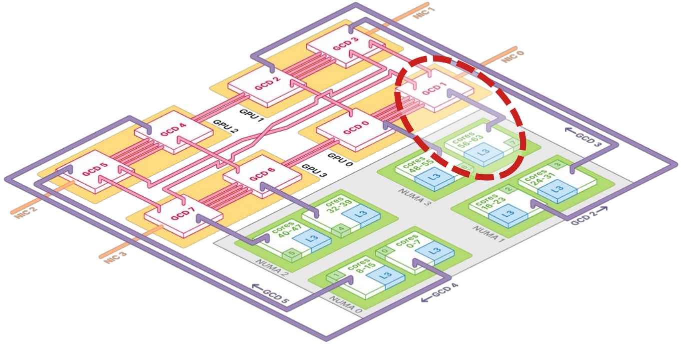
Icons: vecteezy.com

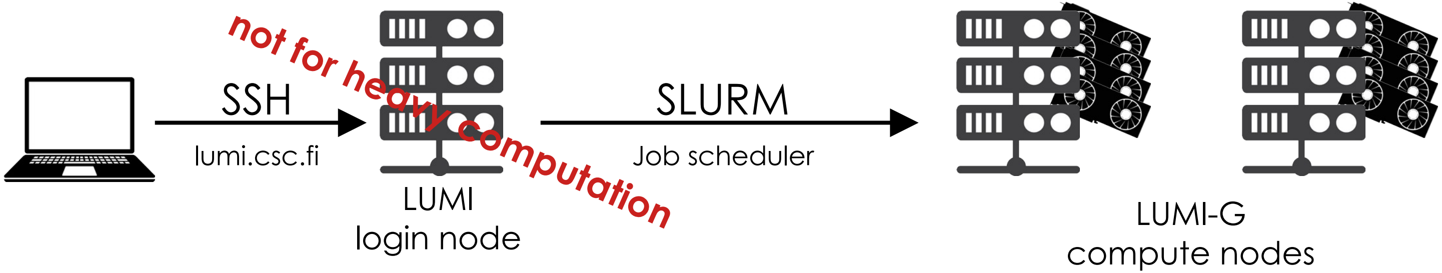
https://docs.lumi-supercomputer.eu/runjobs/

Training on a single Graphics Compute Die



```
$ python my_pytorch_script.py
```





SSH

lumi.csc.fi

not for heavy computation

LUMI
login node

SLURM

Job scheduler

LUMI-G
compute nodes

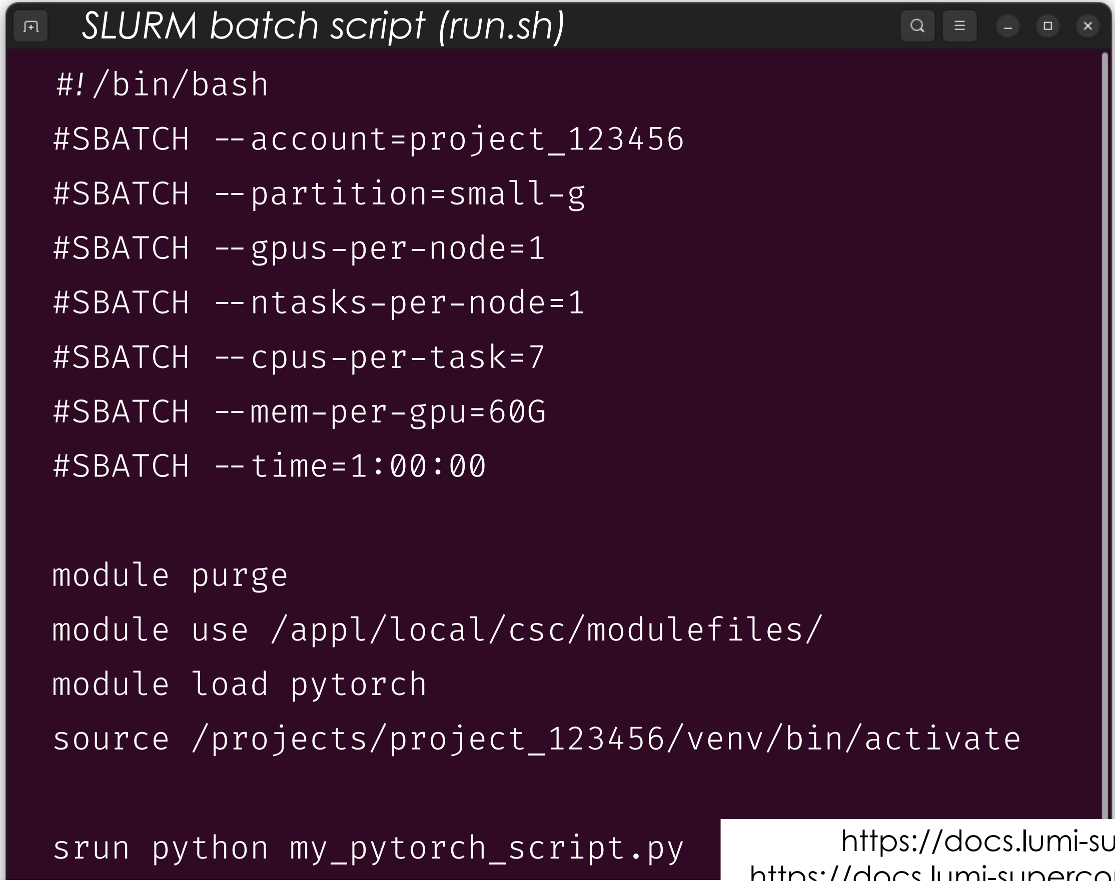
SLURM is used to reserve resources and submit scripts for running on the compute nodes

Icons: vecteezy.com

https://docs.lumi-supercomputer.eu/runjobs/

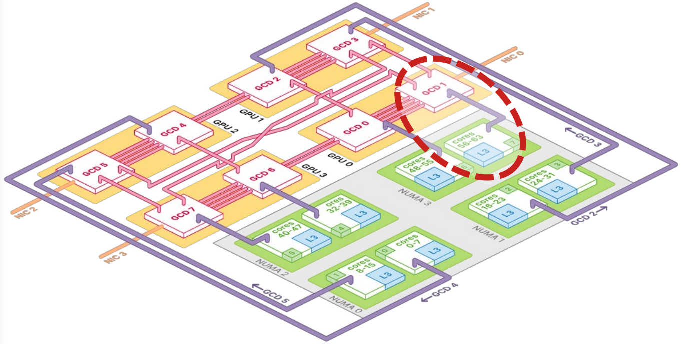
Training on a single Graphics Compute Die

LUMI



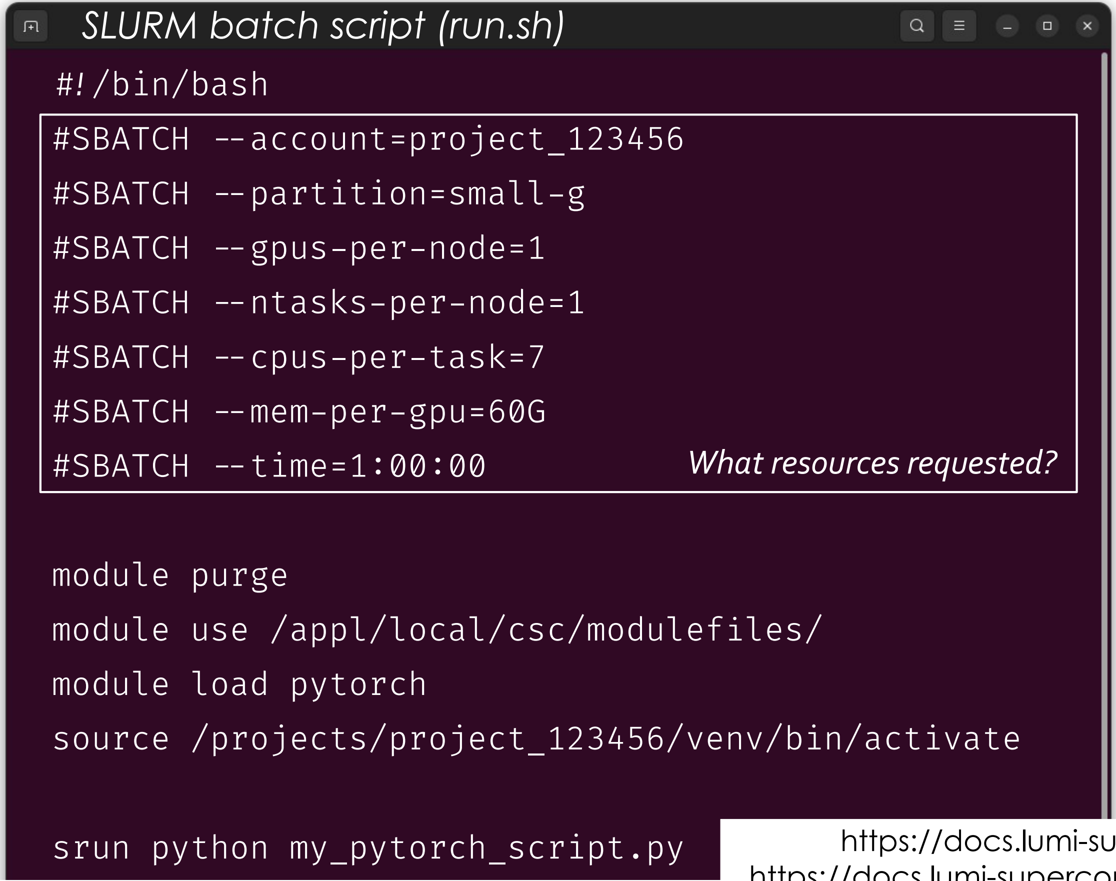
SLURM batch script (run.sh)

```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=small-g
#SBATCH --gpus-per-node=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=7
#SBATCH --mem-per-gpu=60G
#SBATCH --time=1:00:00

module purge
module use /appl/local/csc/modulefiles/
module load pytorch
source /projects/project_123456/venv/bin/activate

srun python my_pytorch_script.py
```



https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/partitions/
https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/slurm-quickstart/

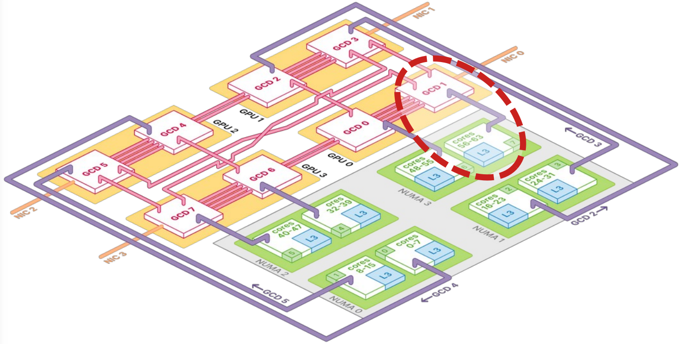
Training on a single Graphics Compute Die



SLURM batch script (run.sh)

```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=small-g
#SBATCH --gpus-per-node=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=7
#SBATCH --mem-per-gpu=60G
#SBATCH --time=1:00:00

module purge
module use /appl/local/csc/modulefiles/
module load pytorch
source /projects/project_123456/venv/bin/activate

srun python my_pytorch_script.py
```

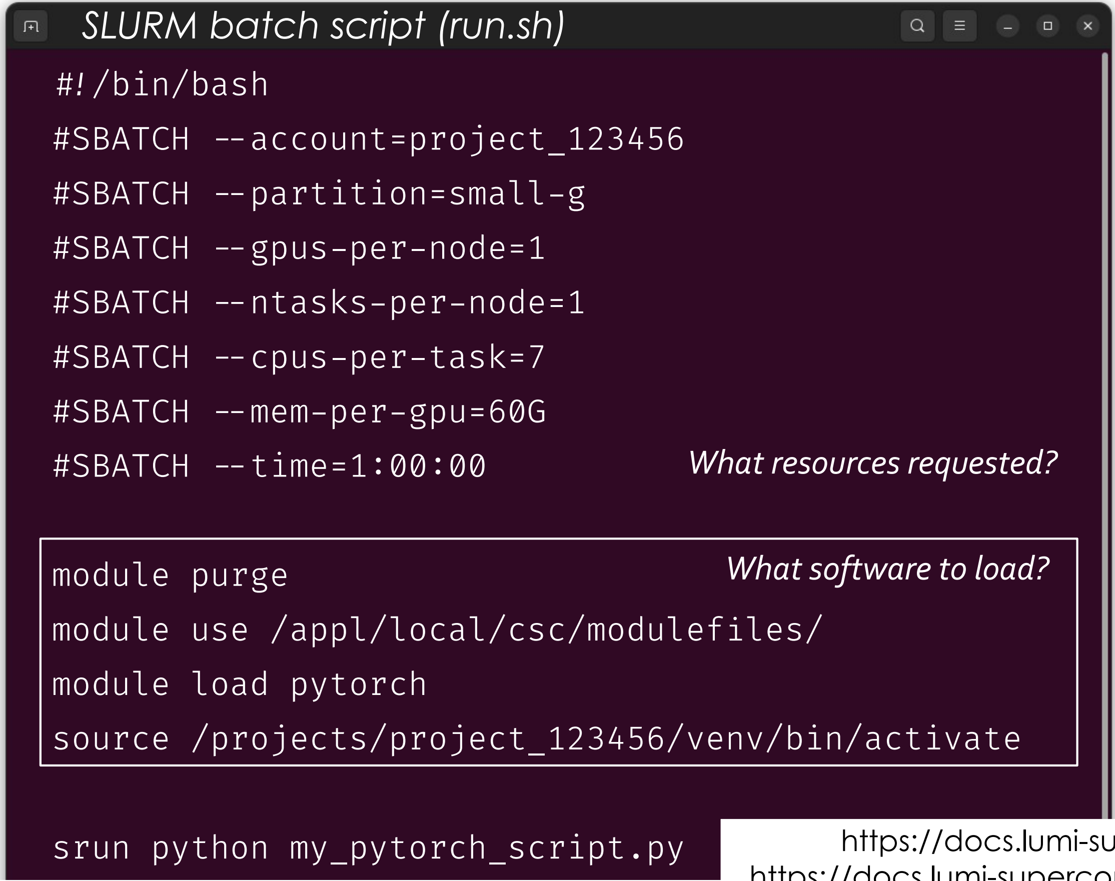
What resources requested?



https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/partitions/
https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/slurm-quickstart/

Training on a single Graphics Compute Die

LUMI



SLURM batch script (run.sh)

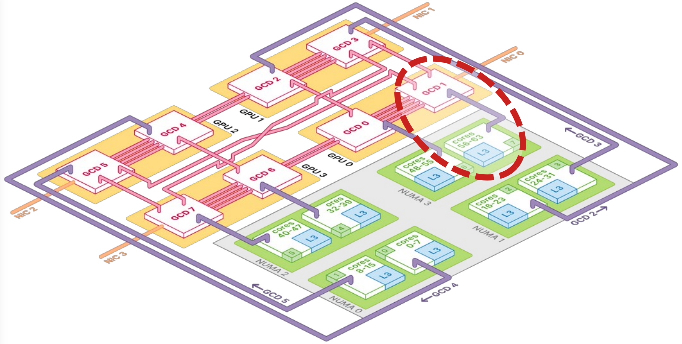
```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=small-g
#SBATCH --gpus-per-node=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=7
#SBATCH --mem-per-gpu=60G
#SBATCH --time=1:00:00
```

What resources requested?

```
module purge
module use /appl/local/csc/modulefiles/
module load pytorch
source /projects/project_123456/venv/bin/activate
```

What software to load?

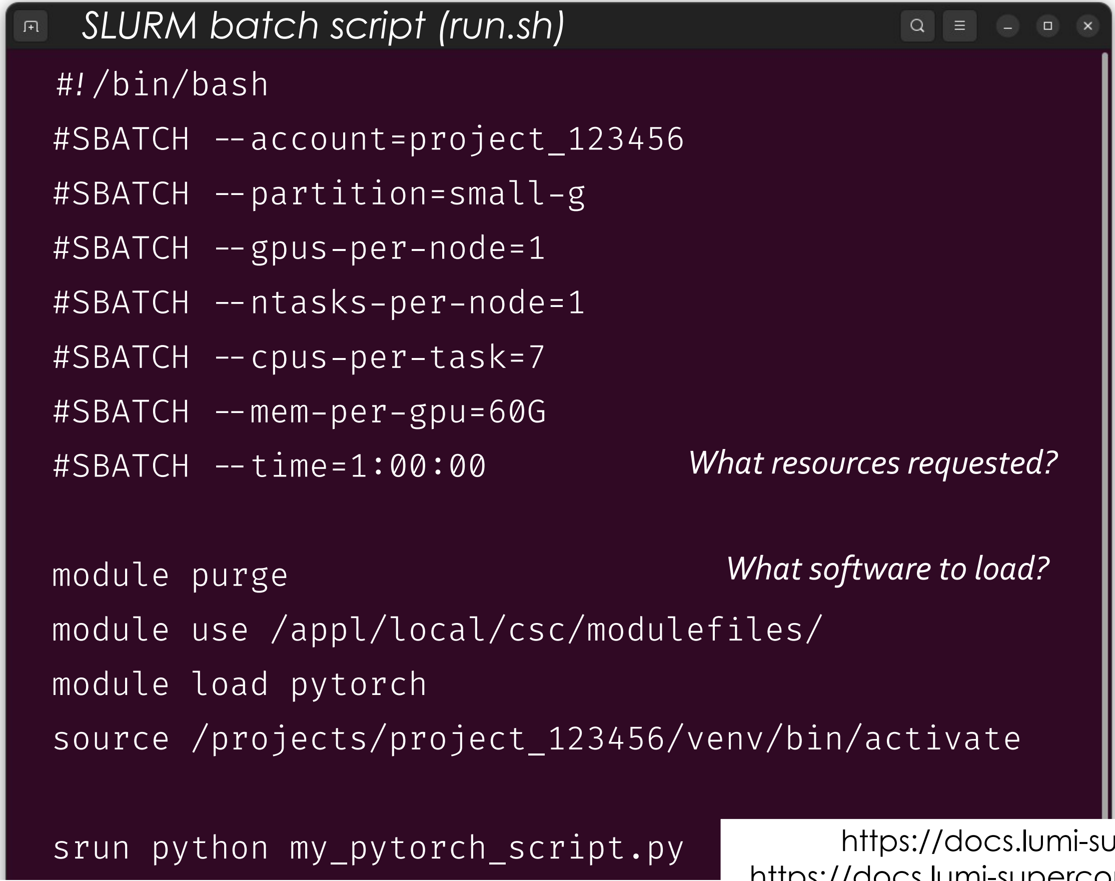
```
srun python my_pytorch_script.py
```



https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/partitions/
https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/slurm-quickstart/

Training on a single Graphics Compute Die

L U M I

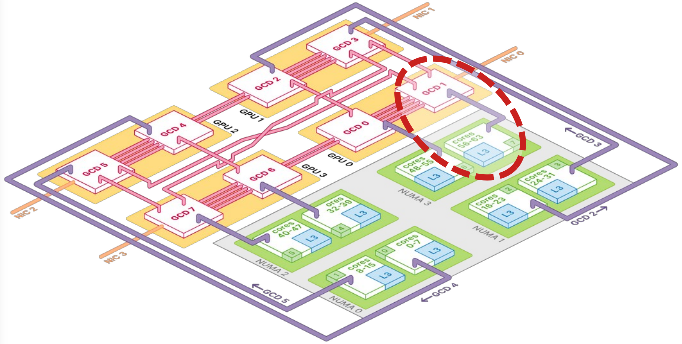


SLURM batch script (run.sh)

```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=small-g
#SBATCH --gpus-per-node=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=7
#SBATCH --mem-per-gpu=60G
#SBATCH --time=1:00:00

module purge
module use /appl/local/csc/modulefiles/
module load pytorch
source /projects/project_123456/venv/bin/activate

srun python my_pytorch_script.py
```

What resources requested?

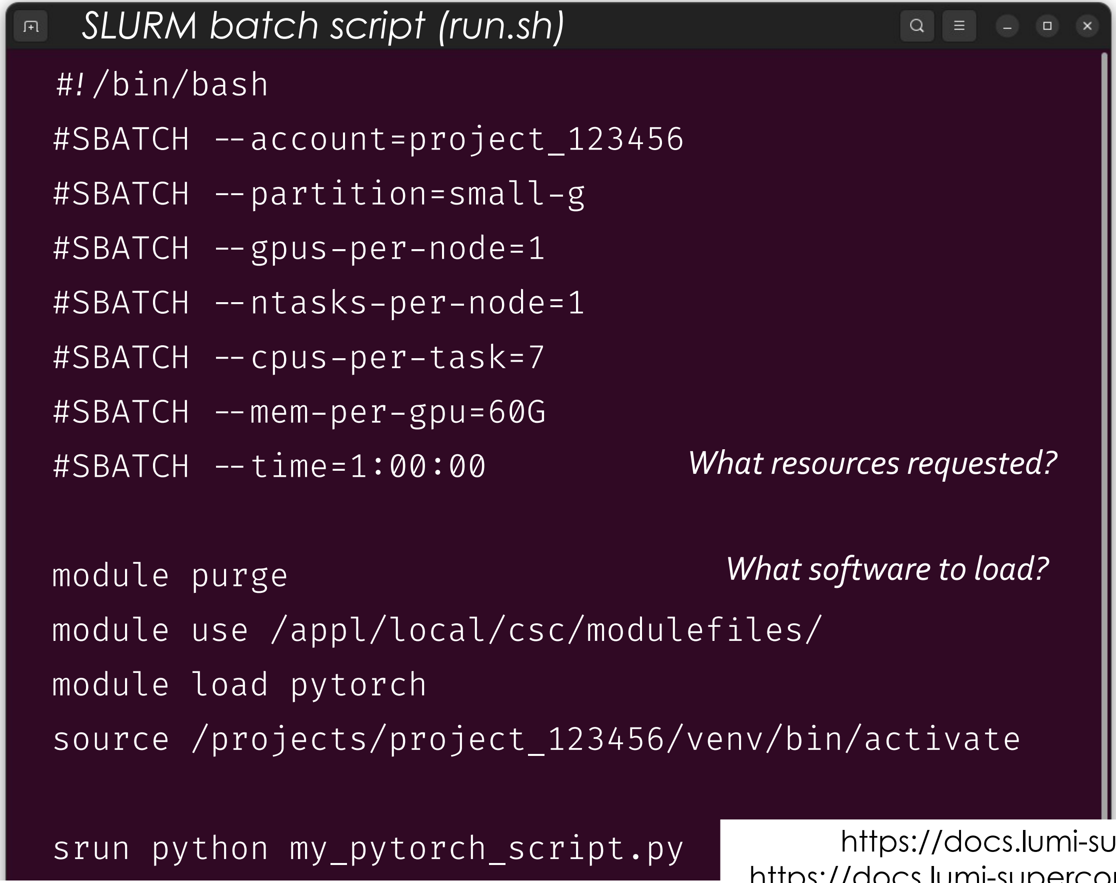
What software to load?



https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/partitions/
https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/slurm-quickstart/

Training on a single Graphics Compute Die

LUMI

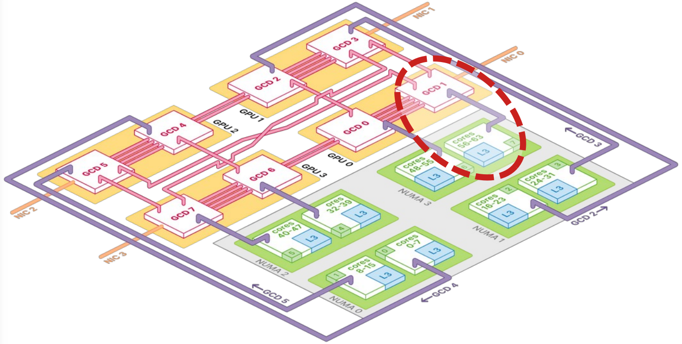


SLURM batch script (run.sh)

```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=small-g
#SBATCH --gpus-per-node=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=7
#SBATCH --mem-per-gpu=60G
#SBATCH --time=1:00:00

module purge
module use /appl/local/csc/modulefiles/
module load pytorch
source /projects/project_123456/venv/bin/activate

srun python my_pytorch_script.py
```

What resources requested?

What software to load?



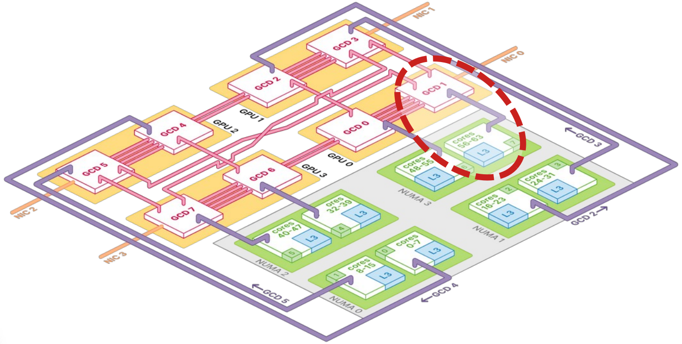
Available GPU partitions

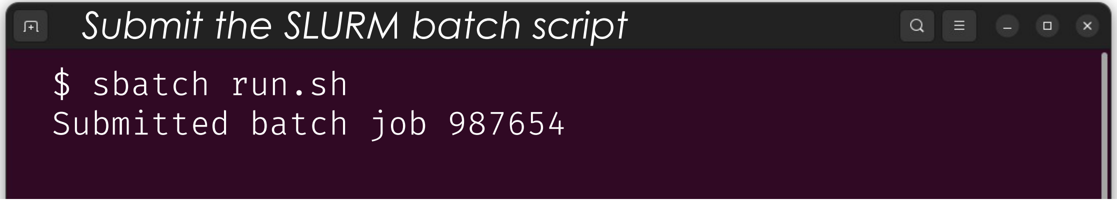
- standard-g
  ≤ 48h, whole nodes only
- small-g
  ≤ 72h, individual GCDs
- dev-g
  ≤ 3h, individual GCDs, 1 job

https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/partitions/
https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/slurm-quickstart/

Training on a single Graphics Compute Die

LUMI



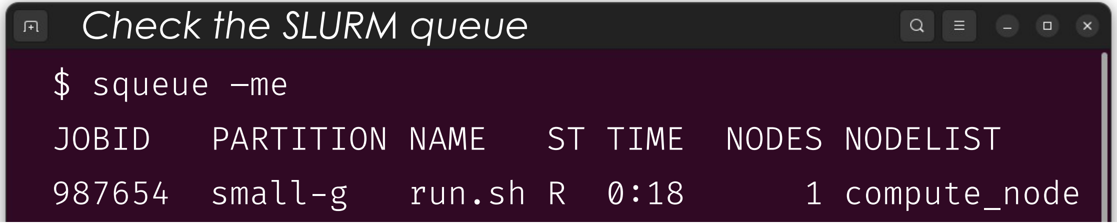


Submit the SLURM batch script

```
$ sbatch run.sh
Submitted batch job 987654
```

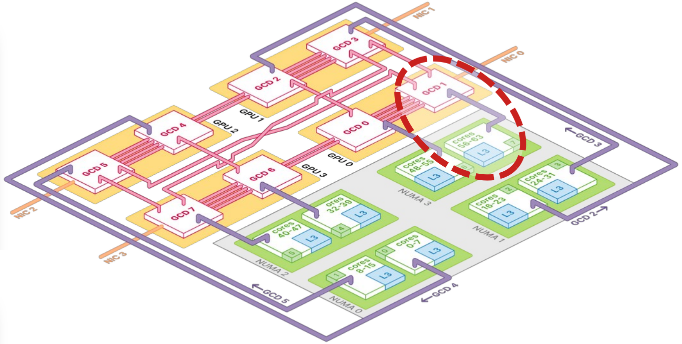
https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/slurm-quickstart/

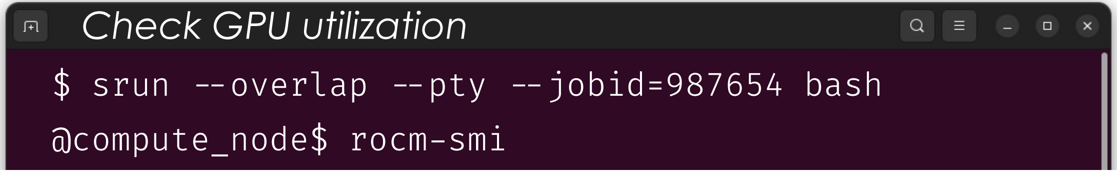
Useful SLURM commands for monitoring job progress



Check the SLURM queue

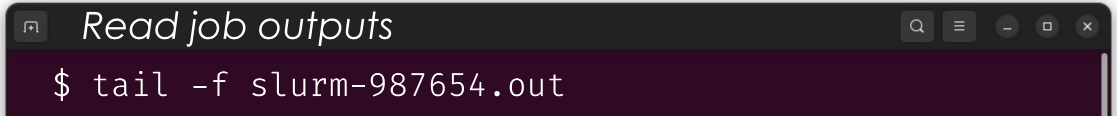
```
$ squeue –me
JOBID   PARTITION NAME   ST TIME  NODES NODELIST
987654  small-g   run.sh R  0:18      1 compute_node
```





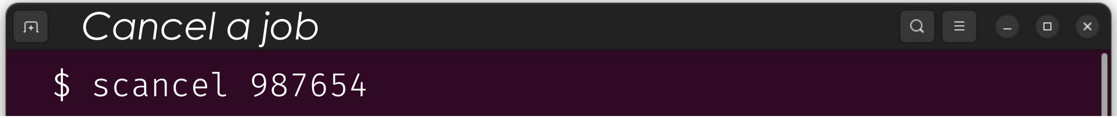
Check GPU utilization

```
$ srun --overlap --pty --jobid=987654 bash
@compute_node$ rocm-smi
```



Read job outputs

```
$ tail -f slurm-987654.out
```



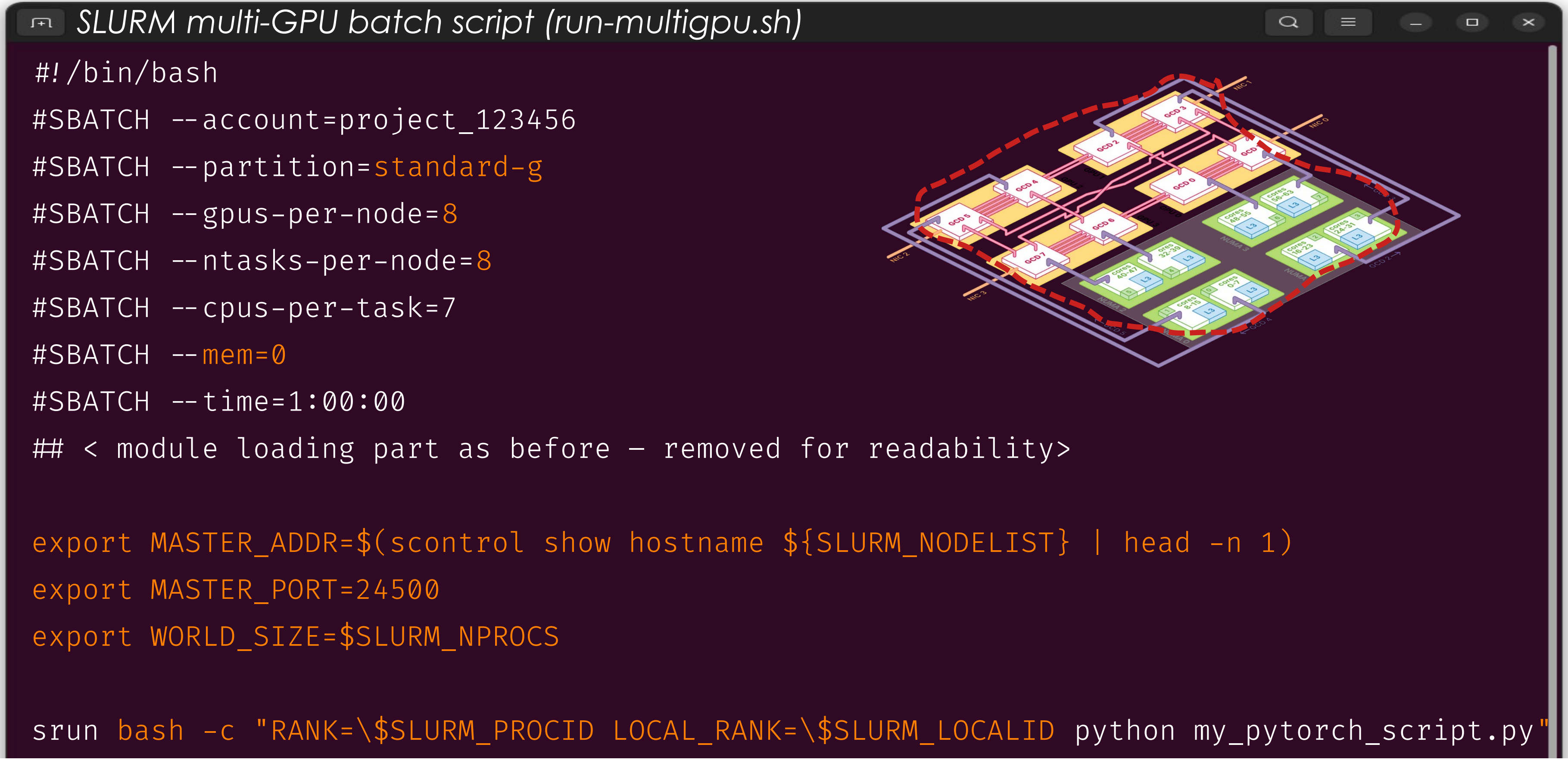
Cancel a job

```
$ scancel 987654
```

https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/slurm-quickstart/

Training on multiple GCDs on a single node

L U M I

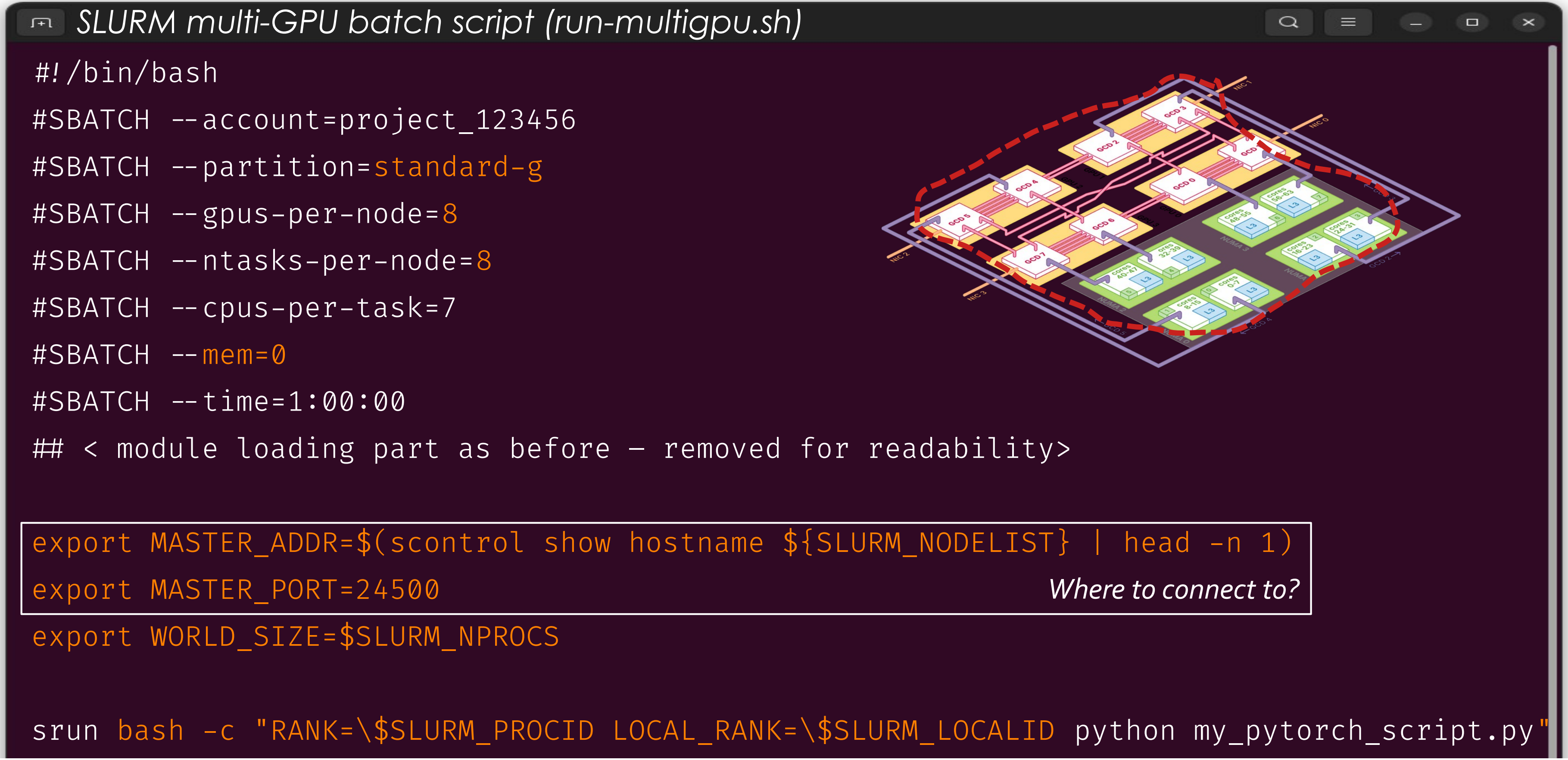


SLURM multi-GPU batch script (run-multigpu.sh)

```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=standard-g
#SBATCH --gpus-per-node=8
#SBATCH --ntasks-per-node=8
#SBATCH --cpus-per-task=7
#SBATCH --mem=0
#SBATCH --time=1:00:00
## < module loading part as before – removed for readability>

export MASTER_ADDR=$(scontrol show hostname ${SLURM_NODELIST} | head -n 1)
export MASTER_PORT=24500
export WORLD_SIZE=$SLURM_NPROCS

srun bash -c "RANK=\$SLURM_PROCID LOCAL_RANK=\$SLURM_LOCALID python my_pytorch_script.py"
```

Training on multiple GCDs on a single node

LUMI



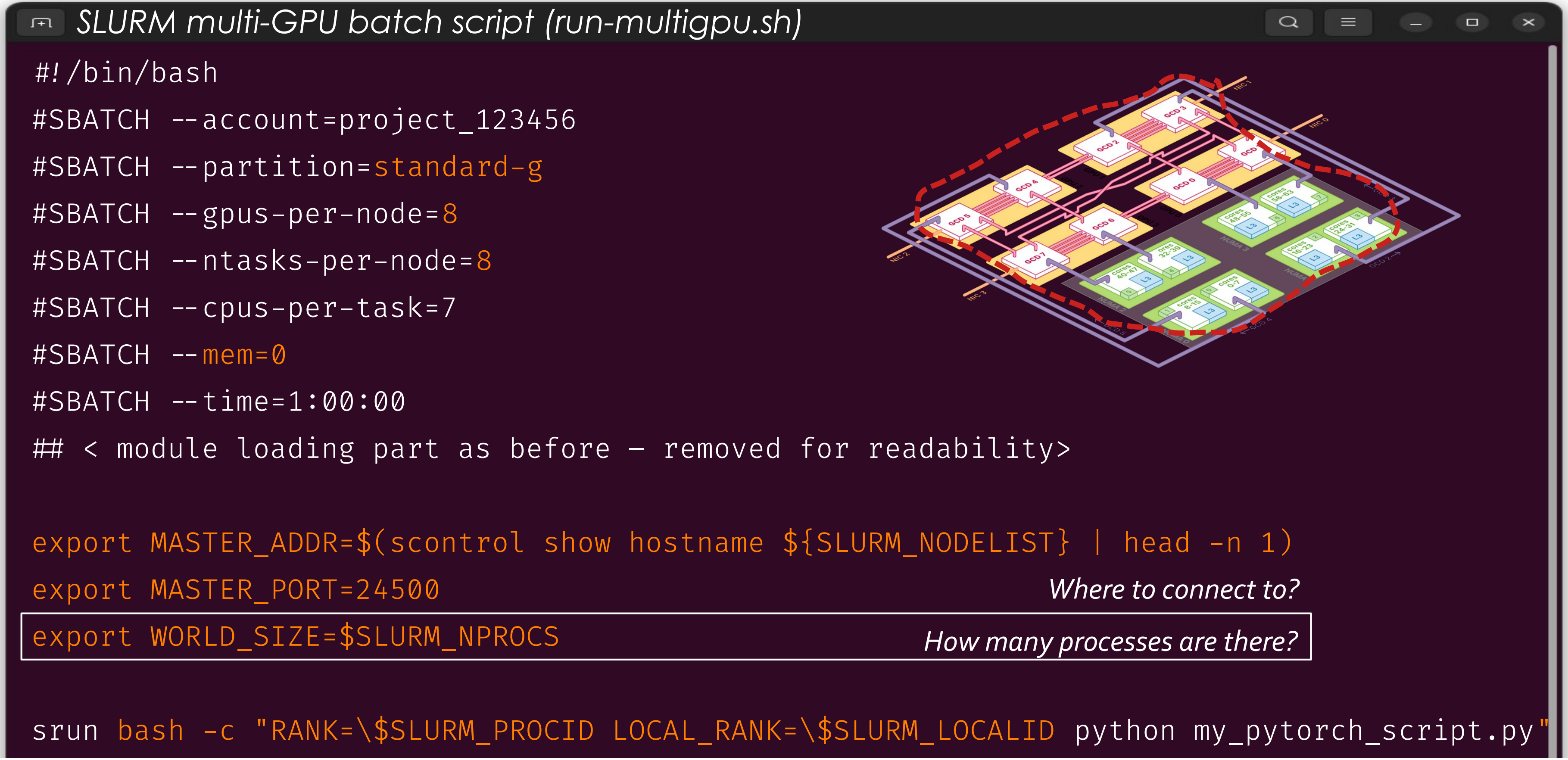
SLURM multi-GPU batch script (run-multigpu.sh)

```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=standard-g
#SBATCH --gpus-per-node=8
#SBATCH --ntasks-per-node=8
#SBATCH --cpus-per-task=7
#SBATCH --mem=0
#SBATCH --time=1:00:00
## < module loading part as before – removed for readability>

export MASTER_ADDR=$(scontrol show hostname ${SLURM_NODELIST} | head -n 1)
export MASTER_PORT=24500
export WORLD_SIZE=$SLURM_NPROCS

srun bash -c "RANK=\$SLURM_PROCID LOCAL_RANK=\$SLURM_LOCALID python my_pytorch_script.py"
```

Where to connect to?

Training on multiple GCDs on a single node

LUMI



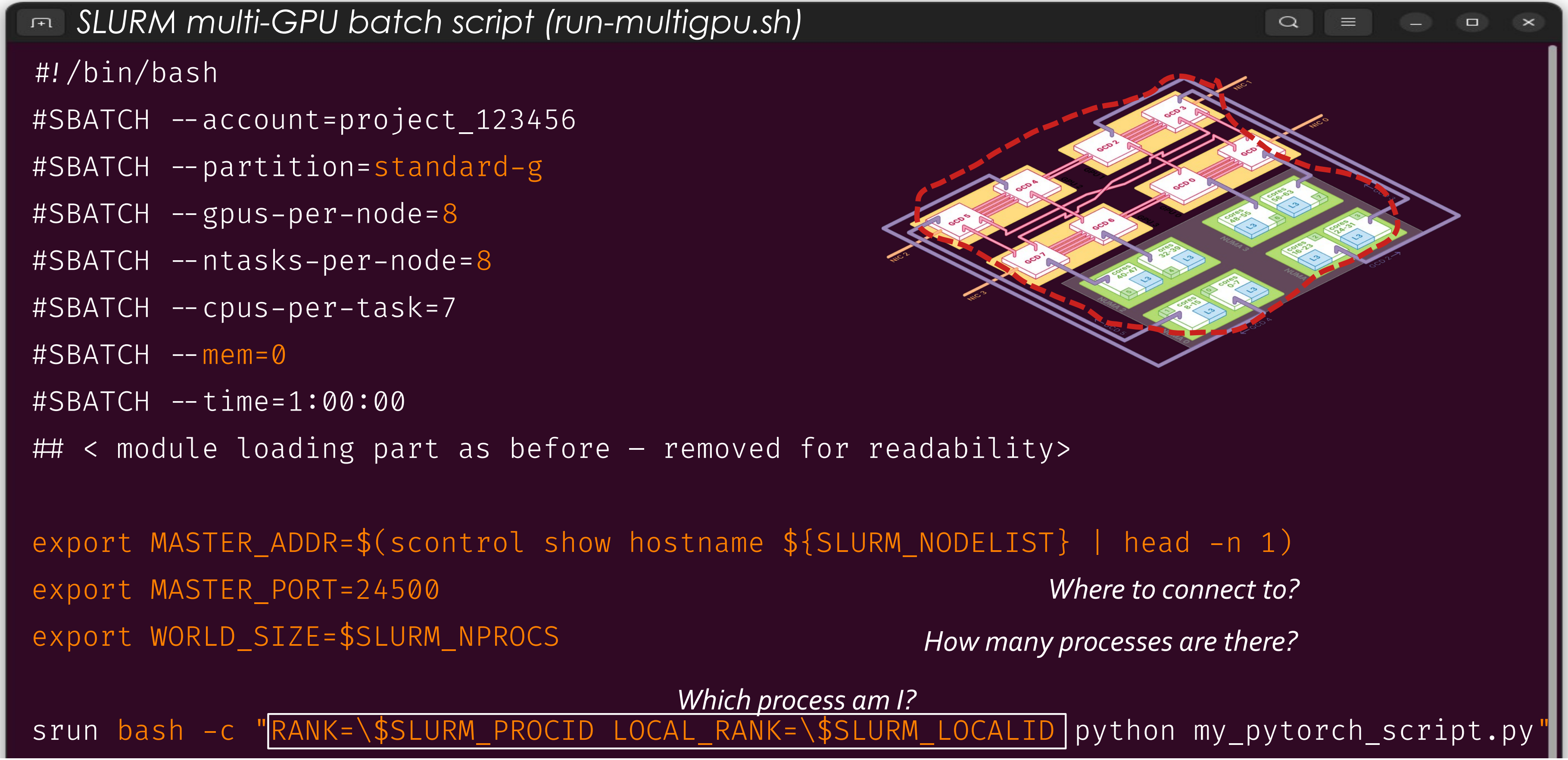
SLURM multi-GPU batch script (run-multigpu.sh)

```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=standard-g
#SBATCH --gpus-per-node=8
#SBATCH --ntasks-per-node=8
#SBATCH --cpus-per-task=7
#SBATCH --mem=0
#SBATCH --time=1:00:00
## < module loading part as before – removed for readability>


export MASTER_ADDR=$(scontrol show hostname ${SLURM_NODELIST} | head -n 1)
export MASTER_PORT=24500
export WORLD_SIZE=$SLURM_NPROCS

srun bash -c "RANK=\$SLURM_PROCID LOCAL_RANK=\$SLURM_LOCALID python my_pytorch_script.py"
```

Where to connect to?

How many processes are there?

Training on multiple GCDs on a single node



SLURM multi-GPU batch script (run-multigpu.sh)

```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=standard-g
#SBATCH --gpus-per-node=8
#SBATCH --ntasks-per-node=8
#SBATCH --cpus-per-task=7
#SBATCH --mem=0
#SBATCH --time=1:00:00
## < module loading part as before – removed for readability>


export MASTER_ADDR=$(scontrol show hostname ${SLURM_NODELIST} | head -n 1)
export MASTER_PORT=24500
export WORLD_SIZE=$SLURM_NPROCS

srun bash -c "RANK=\$SLURM_PROCID LOCAL_RANK=\$SLURM_LOCALID python my_pytorch_script.py"
```

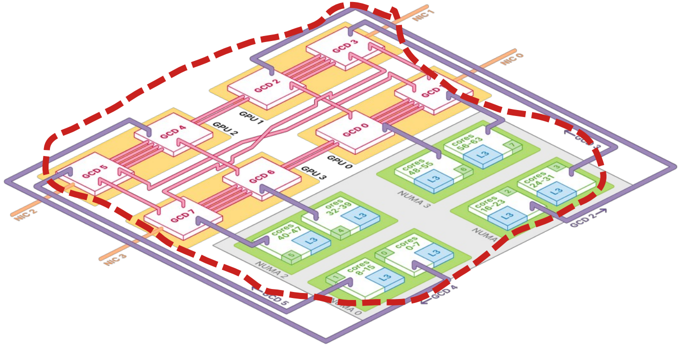
Where to connect to?

How many processes are there?

Which process am I?

Training on multiple GCDs on a single node

Do we need to change the code?



`transformers.Trainer` is automatically set up for distributed training when `WORLD_SIZE` & `RANK` environment variables are set.

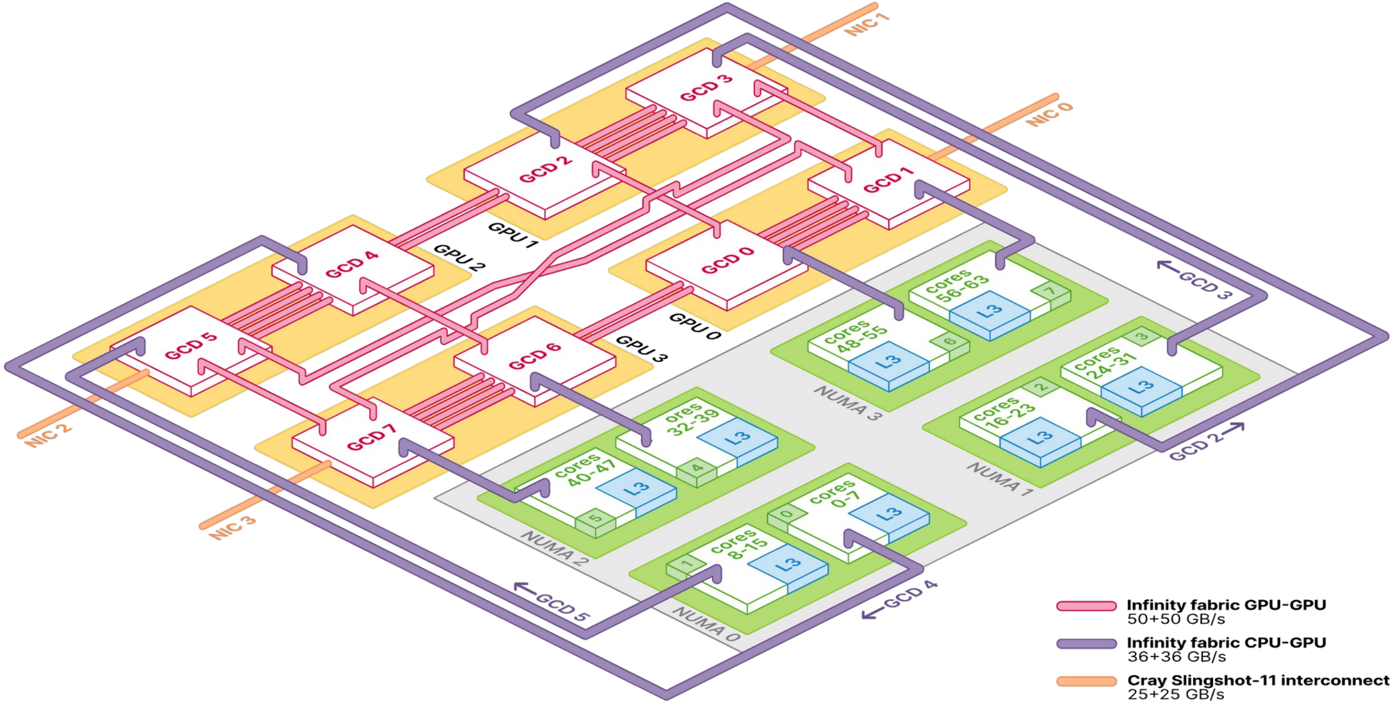
Similar for other frameworks.

But: If using `torch.distributed` directly, add
`dist.init_process_group(backend='nccl')`

Need to read `WORLD_SIZE` & `RANK` to adjust batch size and outputs prints.

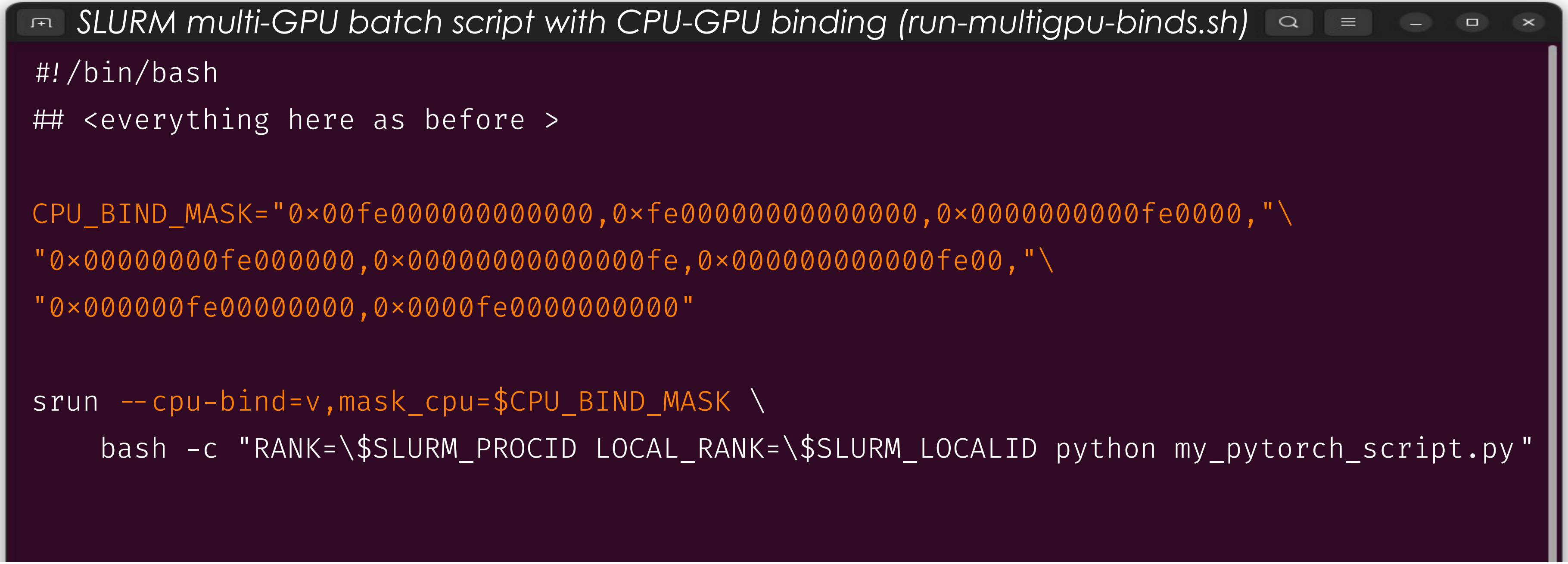
Performance Boost(?): CPU Bindings

LUMI



Performance Boost(?): CPU Bindings

LUMI



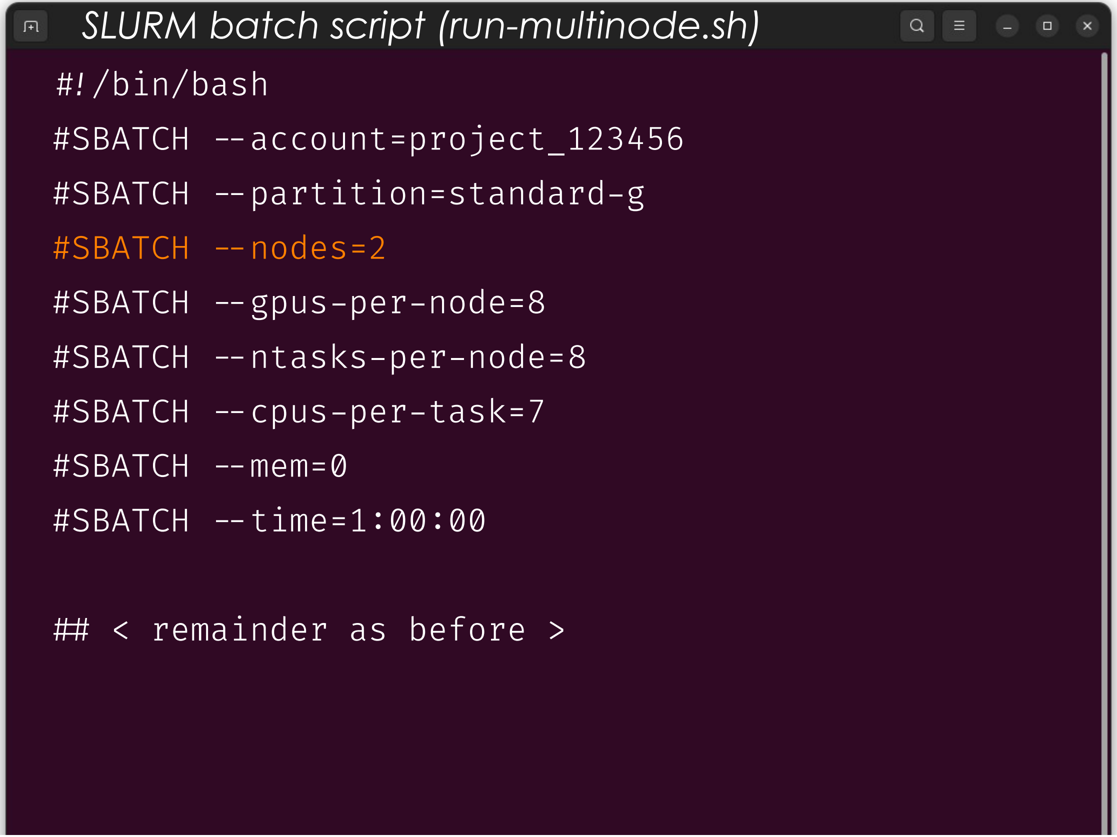
SLURM multi-GPU batch script with CPU-GPU binding (run-multigpu-binds.sh)

```
#!/bin/bash
## <everything here as before >

CPU_BIND_MASK="0x00fe000000000000,0xfe00000000000000,0x0000000000fe0000,"\
"0x00000000fe000000,0x00000000000000fe,0x000000000000fe00,"\
"0x000000fe00000000,0x0000fe0000000000"

srun --cpu-bind=v,mask_cpu=$CPU_BIND_MASK \
    bash -c "RANK=\$SLURM_PROCID LOCAL_RANK=\$SLURM_LOCALID python my_pytorch_script.py"
```

https://docs.lumi-supercomputer.eu/runjobs/scheduled-jobs/distribution-binding/

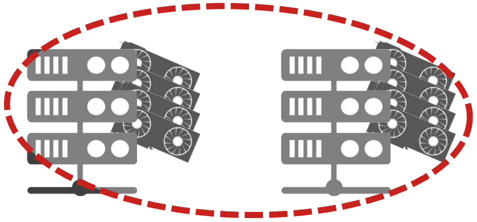
Training on multiple nodes

LUMI



SLURM batch script (run-multinode.sh)

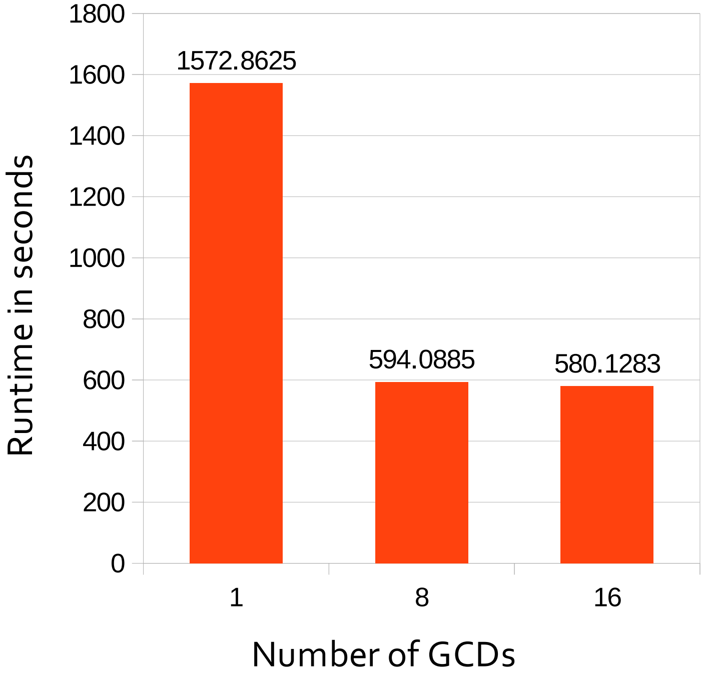
```
#!/bin/bash
#SBATCH --account=project_123456
#SBATCH --partition=standard-g
#SBATCH --nodes=2
#SBATCH --gpus-per-node=8
#SBATCH --ntasks-per-node=8
#SBATCH --cpus-per-task=7
#SBATCH --mem=0
#SBATCH --time=1:00:00


## < remainder as before >
```



Icons: vecteezy.com

Outlook: Scaling more

Data parallelism only gets us so far… Need to consider other scaling “dimensions”



Communication overhead &
GPU underutilization

Outlook: Scaling more

Tensor Parallelism

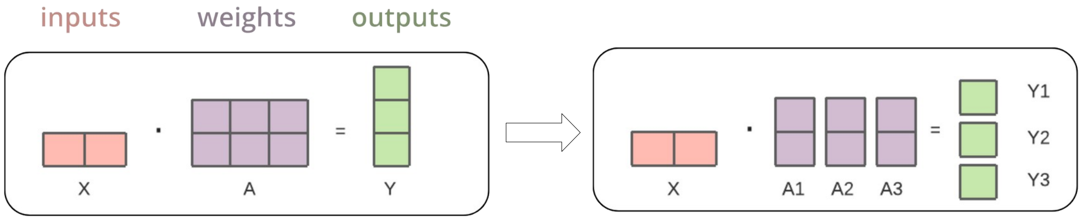


Image adapted from: https://huggingface.co/docs/text-generation-inference/conceptual/tensor_parallelism

deepspeed

Pipeline Parallelism

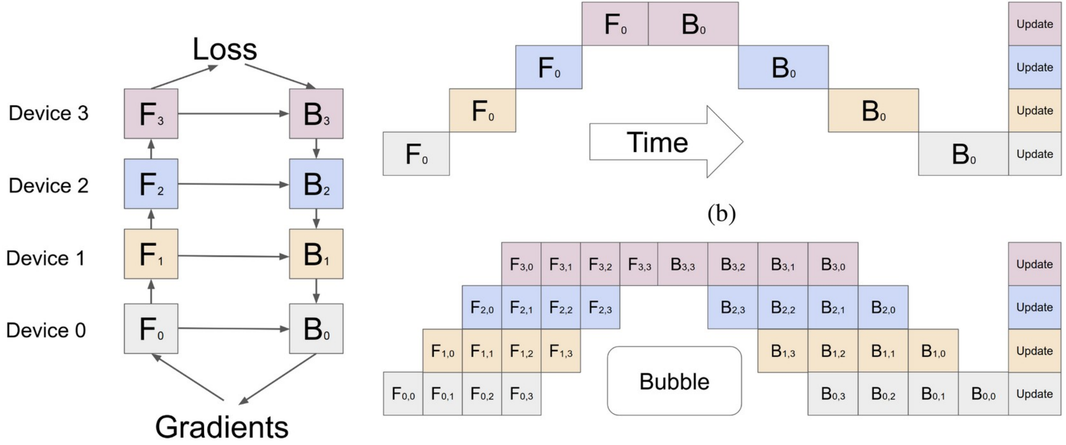

Loss
Device 3
Device 2
Device 1
Device 0
F_3
F_2
F_1
F_0
B_3
B_2
B_1
B_0
Gradients
F_0
B_0
Time
Update
(b)
$F_{3,0}$ $F_{3,1}$ $F_{3,2}$ $F_{3,3}$ $B_{3,3}$ $B_{3,2}$ $B_{3,1}$ $B_{3,0}$
$F_{2,0}$ $F_{2,1}$ $F_{2,2}$ $F_{2,3}$ $B_{2,3}$ $B_{2,2}$ $B_{2,1}$ $B_{2,0}$
$F_{1,0}$ $F_{1,1}$ $F_{1,2}$ $F_{1,3}$ $B_{1,3}$ $B_{1,2}$ $B_{1,1}$ $B_{1,0}$
$F_{0,0}$ $F_{0,1}$ $F_{0,2}$ $F_{0,3}$ $B_{0,3}$ $B_{0,2}$ $B_{0,1}$ $B_{0,0}$
Bubble

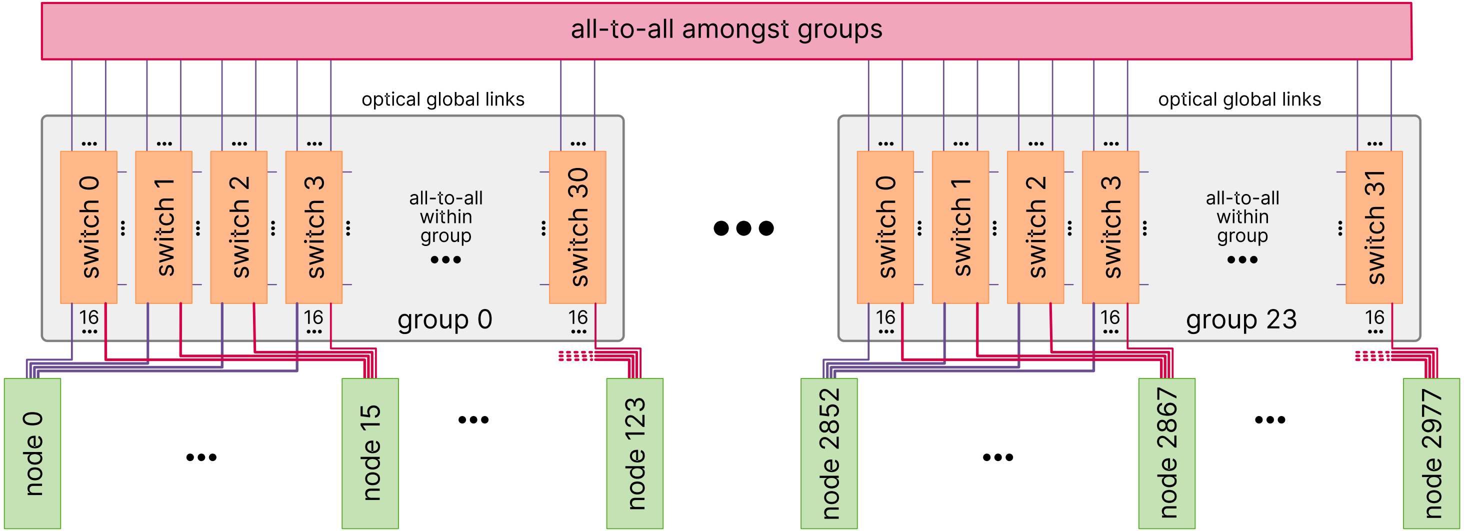
Image: https://arxiv.org/abs/1811.06965

PyTorch Fully Sharded Data Parallel

Outlook: Scaling more

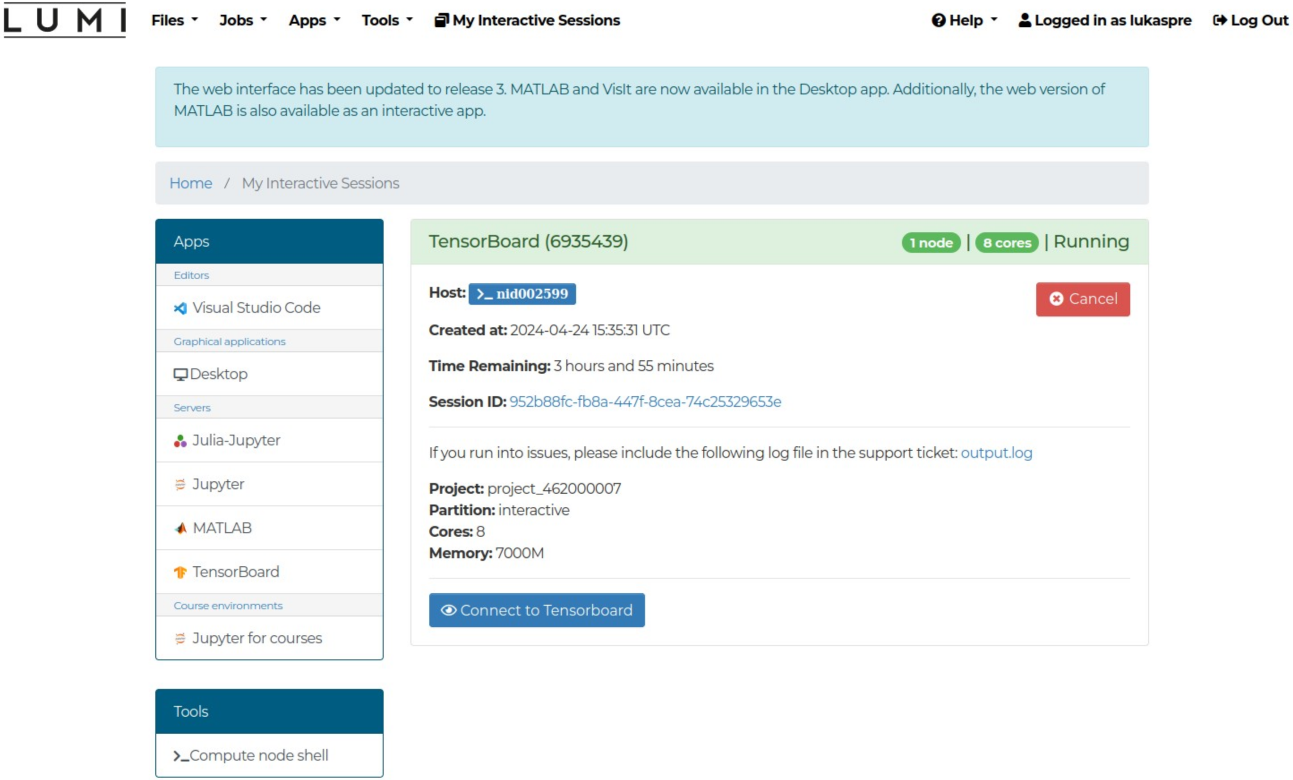
LUMI

For massive scale: Might need to consider network topology



Monitoring Progress

Tensorboard on LUMI web interface: `lumi.csc.fi`



The web interface has been updated to release 3. MATLAB and VisIt are now available in the Desktop app. Additionally, the web version of MATLAB is also available as an interactive app.

Home / My Interactive Sessions

TensorBoard (6935439) 1 node | 8 cores | Running

Host: >_ nid002599

Created at: 2024-04-24 15:35:31 UTC

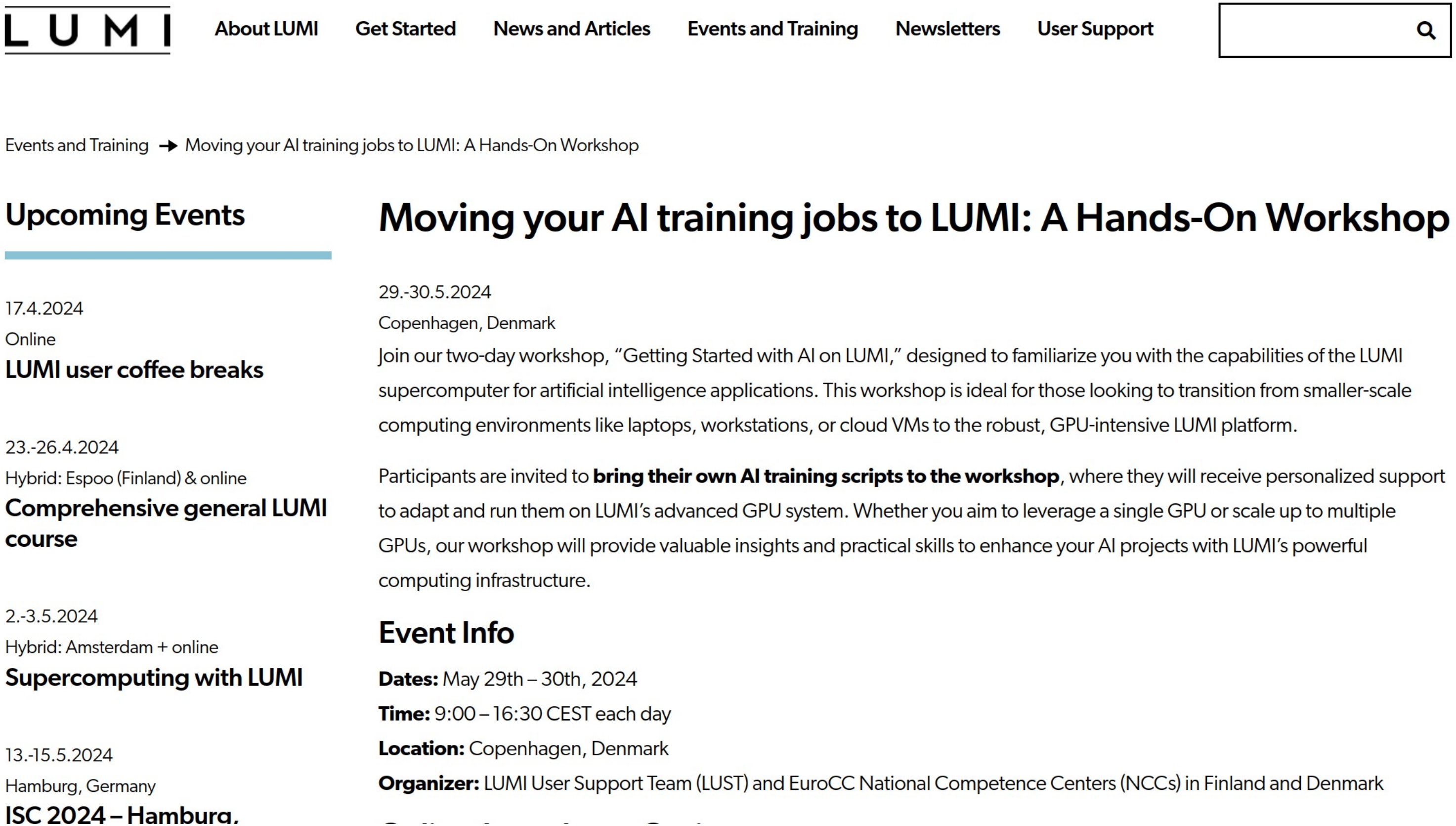
Time Remaining: 3 hours and 55 minutes

Session ID: 952b88fc-fb8a-447f-8cea-74c25329653e

If you run into issues, please include the following log file in the support ticket: output.log

Project: project_462000007
Partition: interactive
Cores: 8
Memory: 7000M

Upcoming LUMI User Support Course



Events and Training → Moving your AI training jobs to LUMI: A Hands-On Workshop

Upcoming Events

17.4.2024
Online
LUMI user coffee breaks

23.-26.4.2024
Hybrid: Espoo (Finland) & online
Comprehensive general LUMI course

2.-3.5.2024
Hybrid: Amsterdam + online
Supercomputing with LUMI

13.-15.5.2024
Hamburg, Germany
ISC 2024 – Hamburg,

Moving your AI training jobs to LUMI: A Hands-On Workshop

29.-30.5.2024
Copenhagen, Denmark

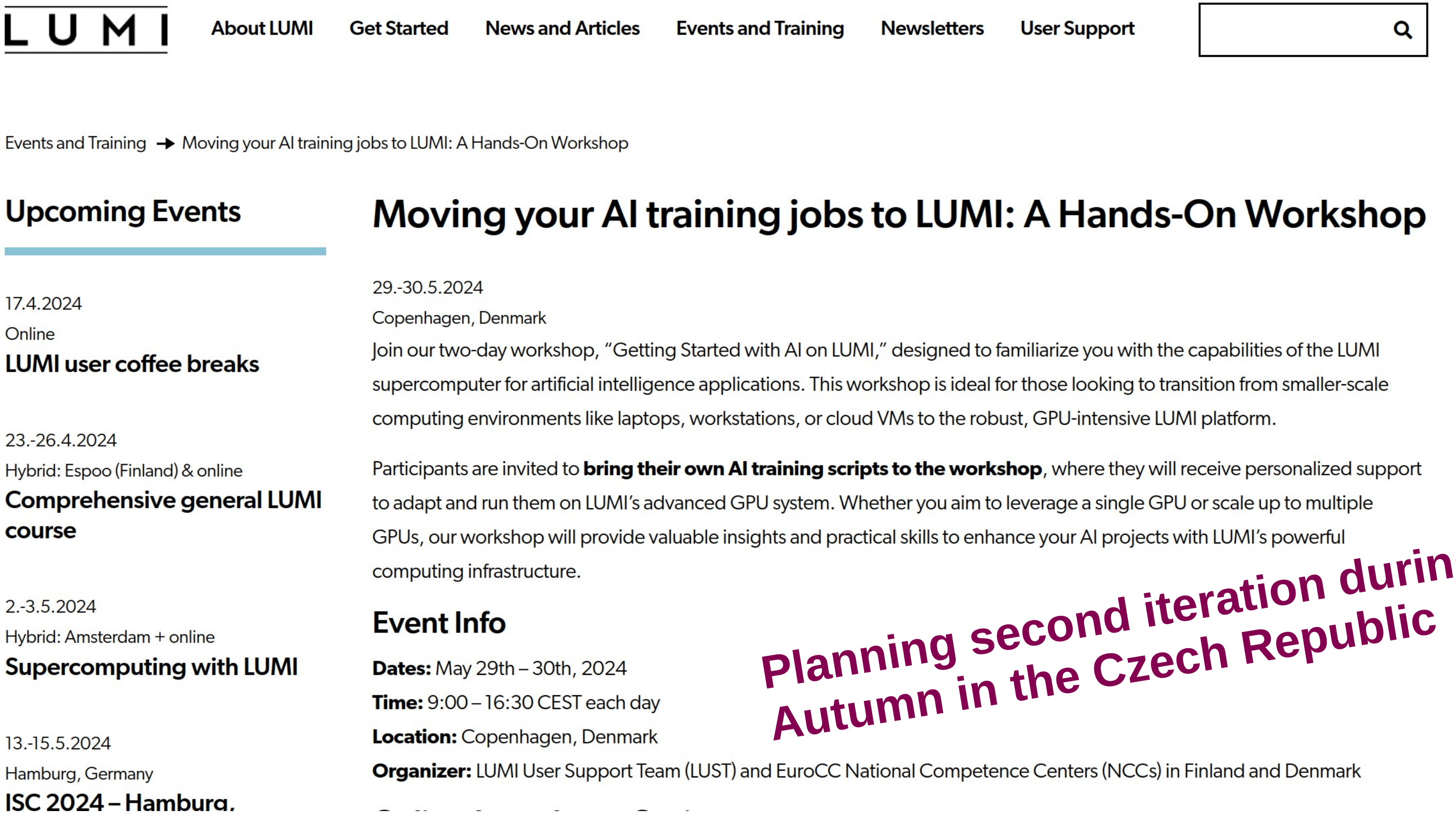
Join our two-day workshop, "Getting Started with AI on LUMI," designed to familiarize you with the capabilities of the LUMI supercomputer for artificial intelligence applications. This workshop is ideal for those looking to transition from smaller-scale computing environments like laptops, workstations, or cloud VMs to the robust, GPU-intensive LUMI platform.

Participants are invited to **bring their own AI training scripts to the workshop**, where they will receive personalized support to adapt and run them on LUMI's advanced GPU system. Whether you aim to leverage a single GPU or scale up to multiple GPUs, our workshop will provide valuable insights and practical skills to enhance your AI projects with LUMI's powerful computing infrastructure.

Event Info

Dates: May 29th – 30th, 2024
Time: 9:00 – 16:30 CEST each day
Location: Copenhagen, Denmark
Organizer: LUMI User Support Team (LUST) and EuroCC National Competence Centers (NCCs) in Finland and Denmark

Upcoming LUMI User Support Course



Events and Training → Moving your AI training jobs to LUMI: A Hands-On Workshop

Upcoming Events

17.4.2024
Online
LUMI user coffee breaks

23.-26.4.2024
Hybrid: Espoo (Finland) & online
Comprehensive general LUMI course

2.-3.5.2024
Hybrid: Amsterdam + online
Supercomputing with LUMI

13.-15.5.2024
Hamburg, Germany
ISC 2024 – Hamburg,

Moving your AI training jobs to LUMI: A Hands-On Workshop

29.-30.5.2024
Copenhagen, Denmark

Join our two-day workshop, "Getting Started with AI on LUMI," designed to familiarize you with the capabilities of the LUMI supercomputer for artificial intelligence applications. This workshop is ideal for those looking to transition from smaller-scale computing environments like laptops, workstations, or cloud VMs to the robust, GPU-intensive LUMI platform.

Participants are invited to **bring their own AI training scripts to the workshop**, where they will receive personalized support to adapt and run them on LUMI's advanced GPU system. Whether you aim to leverage a single GPU or scale up to multiple GPUs, our workshop will provide valuable insights and practical skills to enhance your AI projects with LUMI's powerful computing infrastructure.

Event Info

Dates: May 29th – 30th, 2024
Time: 9:00 – 16:30 CEST each day
Location: Copenhagen, Denmark
Organizer: LUMI User Support Team (LUST) and EuroCC National Competence Centers (NCCs) in Finland and Denmark

Planning second iteration during Autumn in the Czech Republic

You don't need to be a superprogrammer to use a supercomputer

LUMI

More Information? Check LUMI docs!
→ https://docs.lumi-supercomputer.eu/
and the links throughout the presentation

CSC PyTorch Module docs
→ https://docs.csc.fi/apps/pytorch/

Want help from a human?
→ https://lumi-supercomputer.eu/user-support/need-help/

Talk to me?
→ here at the conference!
→ lukas.prediger@csc.fi