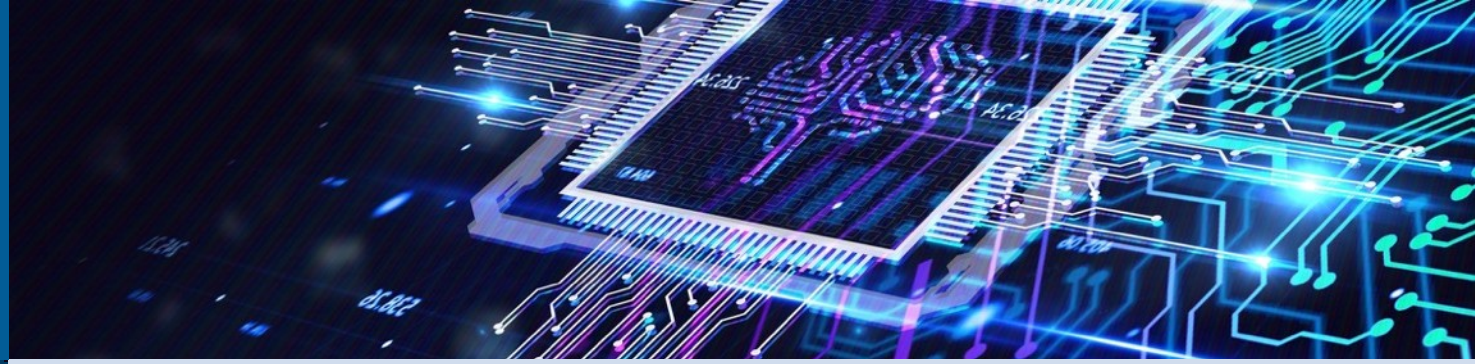
CSC

ICT Solutions for Brilliant Minds



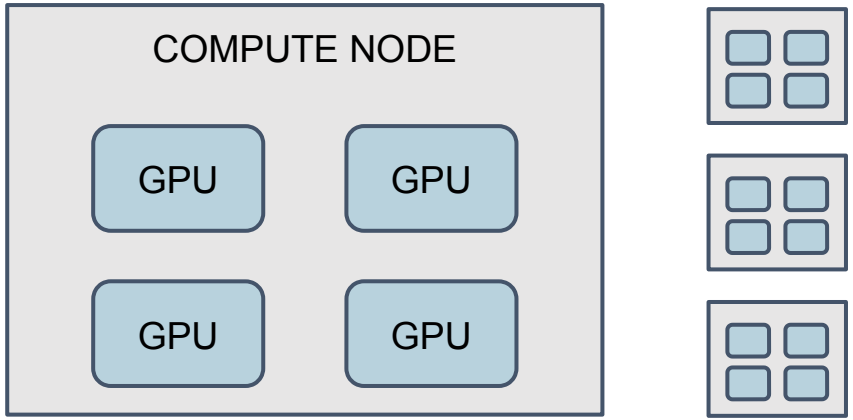
Machine learning on supercomputers, Part 3: Multi-GPU and multi-node jobs

Mats Sjöberg, CSC

Using multiple GPUs

Each node (computer) has

- Puhti and Mahti: 4 GPUs
- LUMI: 8 GPUs
  (actually 4 dual chip GPUs)



Minimum number of nodes:

- if you need 1-4 GPUs (1-8 in LUMI) reserve a single node only

```
--nodes=1
--gres=gpu:v100:4
```

- if you need more, reserve in multiples of 4 (8 in LUMI)

```
--nodes=2
--gres=gpu:v100:4
```

Make sure you can actually use multiple GPUs!

- **Simply reserving more GPUs is not enough!**
- Your code or software framework needs to explicitly support it
- Check with seff or nvidia-smi/rocm-smi if you are unsure

```
GPU job efficiency:
------------------------------------------------------------------------
GPU load
      Hostname        GPU Id      Mean (%)     stdDev (%)        Max (%)
        r02g01             0         75.56          42.84          98.00
        r02g01             1         75.22          42.65          97.00
        r02g01             2         75.44          42.77          97.00
        r02g01             3         75.44          42.77          97.00
```

https://docs.csc.fi/support/tutorials/gpu-ml/#gpu-utilization

Reminder: single GPU on Puhti

```
#!/bin/bash
#SBATCH --account=<project>
#SBATCH --partition=gpu
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=10
#SBATCH --mem=64G
#SBATCH --time=1:00:00
#SBATCH --gres=gpu:v100:1

srun python3 myprog.py <options>
```

Rule of thumb: reserve other resources in proportion of GPUs (or less)

Puhti node has:

- 4 GPUs
- 40 CPU cores
  → max 10 cores/GPU
- 382 000 MB RAM
  → max 95500 MB/GPU or
  ~ 93GB/GPU

Multiple GPUs, using MPI tasks or not?

- Typically we allocate a separate CPU process for each GPU
- Two common solutions:
  - Software framework handles the starting of multiple processes
    Example: PyTorch DDP (uses elastic/rendezvous)
  - Use MPI tasks to start multiple processes
    Example: PyTorch Lightning and Horovod,
    DeepSpeed can use it if configured to
- **You need to know what approach your framework uses!**
- Frameworks should use NCCL (Puhti, Mahti) or RCCL (LUMI) for fast inter-GPU communication

Single node on Puhti: *no* MPI

```
#!/bin/bash
#SBATCH --account=<project>
#SBATCH --partition=gpu
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=40
#SBATCH --mem=320G
#SBATCH --time=1:00:00
#SBATCH --gres=gpu:v100:4

srun python3 myprog.py <options>
```

Here we reserve a full single node:

- 4 GPUs
- 10 x 4 = 40 CPU cores
- Max 373 GB of memory
- No MPI, i.e., single task

Single node on Puhti: *with* MPI

```
#!/bin/bash
#SBATCH --account=<project>
#SBATCH --partition=gpu
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=4
#SBATCH --cpus-per-task=10
#SBATCH --mem=320G
#SBATCH --time=1:00:00
#SBATCH --gres=gpu:v100:4

srun python3 myprog.py <options>
```

Again a full node of 4 GPUs:

- 4 MPI tasks for 4 GPUs
- 10 CPU cores *per task*
- Max 373 GB of memory *for the whole job*
- **Note:** line starting with *srun* will be launched 4 times
  - if your job doesn't understand MPI it will run 4 identical jobs!

Multiple nodes: example 2 nodes → 8 GPUs

No MPI

```
#!/bin/bash
#SBATCH --account=<project>
#SBATCH --partition=gpu
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=40
#SBATCH --mem=320G
#SBATCH --time=1:00:00
#SBATCH --gres=gpu:v100:4

srun python3 myprog.py <options>
```

With MPI

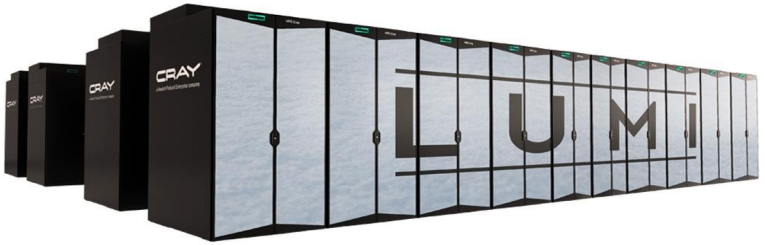
```
#!/bin/bash
#SBATCH --account=<project>
#SBATCH --partition=gpu
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=4
#SBATCH --cpus-per-task=10
#SBATCH --mem=320G
#SBATCH --time=1:00:00
#SBATCH --gres=gpu:v100:4

srun python3 myprog.py <options>
```

Machine learning guide in docs.csc.fi

https://docs.csc.fi/support/tutorials/ml-multi/

- Multi-GPU and multi-node tutorials for:
  - PyTorch DistributedDataParallel
  - PyTorch Lightning
  - DeepSpeed
- Code examples: https://github.com/CSCfi/pytorch-ddp-examples

Using *a lot of* GPUs on LUMI



yle Etusivu Vaalikone Venäjän hyökkäys UMK24

Tiede

FinGPT3 on suurin puhtaasti suomenkielinen kielimalli, eikä suurempaa ole hetkeen tulossa

Uutinen

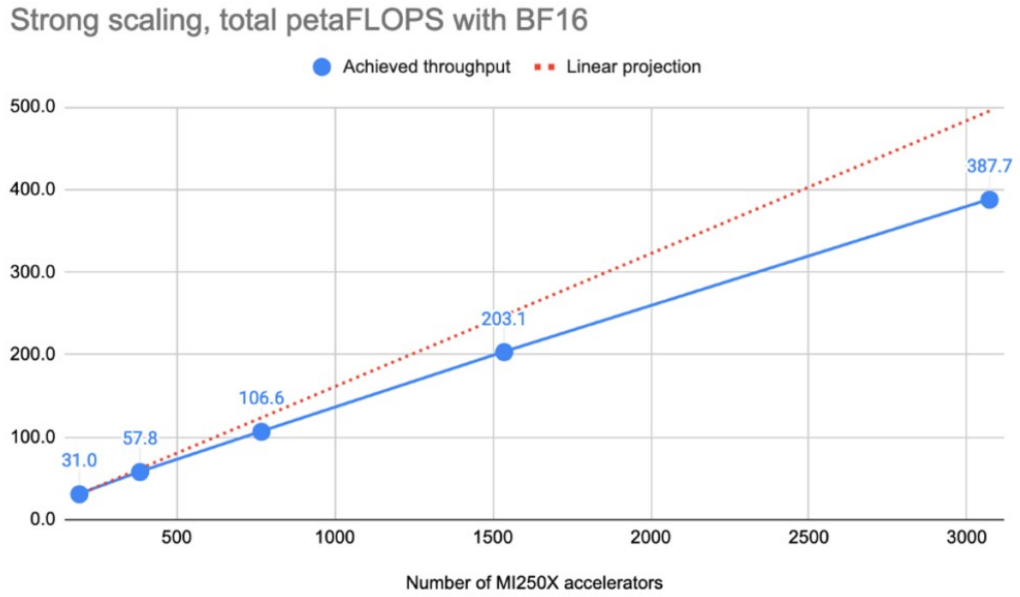
A New Foundation for AI Is Being Built in Finland – Offering an Alternative to American Giants

AI2 OLMo
Open Language Model

News

2.2.2024

A truly open large language model released, developed with LUMI



Sources: Yle, Tivi, LUMI web site