Podpora AI, NVIDIA cloud a jeho použití s pomocí GPU

Jan Hoidekr

AI, NGC, GPGPU

- GPU in Metacentrum/CERIT-SC
  - Hardware and PBS

- NVIDIA GPU CLOUD
  - Examples for AI

GPU in Metacentum

- PBS queues
  - gpu@meta-pbs.cesnet.cz
  - gpu@cerit-pbs.cerit-sc.cz
  - *migration between queues*!
- PBS resources
  - ngpus
  - gpu_cap
    - CUDA compute capability
    - cuda35 – cuda80
- CUDA, drivers – 11.2, 460.39
- A100 – TensorCores

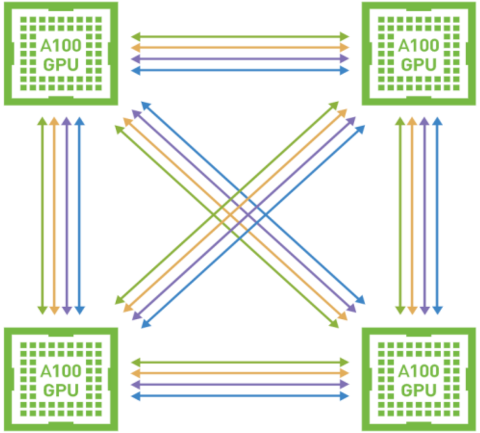
Cluster	GPU model	Cores/mem/cap
Doom, Zubat	Tesla K20m	2496/ 5GB/35
Black1, Grimbold	Tesla P100	3584/16GB/60
Konos	GTX 1080 Ti	3548/11GB/61
Glados	TITAN V	5120/12GB/70
Fau	Q RTX 5000	3072/16GB/75
Adan	Tesla T4	2650/16GB/75
Cha, Gita	RTX 2080 Ti	4352/11GB/75
Zia	A100	6912/40GB/80

GPU in Metacentum

- PBS job – reservation of GPU(s), not shared use
  - do not touch CUDA_VISIBLE_DEVICES env !

- singleNode/singleGPU -> multiNode/multiGPU
  - NCCL library
  - A100 with NVLink

- Our plans
  - new GPU cluster this year
  - PBS resource – gpu_mem



AI, Machine Learning, Deep Learning

- Popular frameworks
  - TensorFlow, PyTorch, Caffe and many others
  - Python modules
    - many requirements and dependencies
    - many modules for specific tasks in different versions
    - quick development

-> Hard to satisfy requirements with module system in Metacentrum

solution with containers – NVIDIA GPU CLOUD

NVIDIA GPU CLOUD

- **Set of GPU-optimized software for AI, HPC and Visualization**

- https://ngc.nvidia.com
- Free registration to full access + Terms of Use

- Docker images
  - TensorFlow, PyTorch and many others
  - updated every month
  - excellent documentation and examples
  - /storage/singularity.metacentrum.cz/NGC – saved singularity images

PyTorch Example

- PyTorch Examples https://github.com/pytorch/examples

```
$ qsub -I -q gpu -l select=1:ncpus=2:ngpus=1:mem=64gb:gpu_cap=cuda80:scratch_local=100gb -l walltime=8:00:00

zia5$ singularity shell --nv -B $SCRATCHDIR /cvmfs/singularity.metacentrum.cz/NGC/PyTorch\:21.03-py3.SIF
Singularity> cd $SCRATCHDIR
Singularity> git clone https://github.com/pytorch/examples.git
Singularity> cd examples/word_language_model/
Singularity> time python main.py --cuda --emsize 650 --nhid 650 --dropout 0.5 --epochs 40 --batch_size=1596
-----------------------------------------------------------------------------------------
| end of epoch   1 | time: 11.32s | valid loss  7.82 | valid ppl  2480.90
< … shortened … >
| End of training | test loss  5.61 | test ppl   273.84
=========================================================================================
real    9m43.756s
user    5m19.087s
sys     4m23.170s
Singularity> exit
zia5$
```

PyTorch Example

- PyTorch Examples https://github.com/pytorch/examples

```
$ qsub -I -q gpu -l select=1:ncpus=2:ngpus=1:mem=64gb:gpu_cap=cuda80:scratch_local=100gb -l walltime=8:00:00

zia5$ singularity shell --nv -B $SCRATCHDIR /cvmfs/singularity.metacentrum.cz/NGC/PyTorch\:21.03-py3.SIF
Singularity> cd $SCRATCHDIR
Singularity> git clone https://github.com/pytorch/examples.git
Singularity> cd examples/word_language_model/
Singularity> time python main.py --cuda --emsize 650 --nhid 650 --dropout 0.5 --epochs 40 --batch_size=1596
-----------------------------------------------------------------------------------------
| end of epoch   1 | time: 11.32s | valid loss  7.82 | valid ppl  2480.90
< … shortened … >
| End of training | test loss  5.61 | test ppl   273.84
=========================================================================================
real    9m43.756s
user    5m19.087s
sys     4m23.170s
Singularity> exit
zia5$
```

PyTorch Example

- PyTorch Examples https://github.com/pytorch/examples

```
$ qsub -I -q gpu -l select=1:ncpus=2:ngpus=1:mem=64gb:gpu_cap=cuda80:scratch_local=100gb -l walltime=8:00:00

zia5$ singularity shell --nv -B $SCRATCHDIR /cvmfs/singularity.metacentrum.cz/NGC/PyTorch\:21.03-py3.SIF
Singularity> cd $SCRATCHDIR
Singularity> git clone https://github.com/pytorch/examples.git
Singularity> cd examples/word_language_model/
Singularity> time python main.py --cuda --emsize 650 --nhid 650 --dropout 0.5 --epochs 40 --batch_size=1596
-----------------------------------------------------------------------------------------
| end of epoch   1 | time: 11.32s | valid loss  7.82 | valid ppl  2480.90
< … shortened … >
| End of training | test loss  5.61 | test ppl   273.84
=========================================================================================
real    9m43.756s
user    5m19.087s
sys     4m23.170s
Singularity> exit
zia5$
```

- NVIDIA_TF32_OVERRIDE=0 disables TF32/TensorCore -> `real    19m28.945s`
- Tesla T4, batch_size=399 -> `real    89m37.136s`

TensorFlow example

- Transformers benchmark
  - Python modules not in image – `transformers py3nvml`
  - **pip** – uses ~/.local/lib/ outside the image (see docs of PYTHONUSERBASE)

```
$ qsub -I -q gpu -l select=1:ncpus=2:ngpus=1:mem=64gb:gpu_cap=cuda80:scratch_local=100gb -l walltime=8:00:00

zia5$ singularity shell --nv -B $SCRATCHDIR /cvmfs/singularity.metacentrum.cz/NGC/TensorFlow\:21.03-tf2-py3.SIF
Singularity> pip install transformers py3nvml
Successfully installed filelock-3.0.12 joblib-1.0.1 py3nvml-0.2.6 regex-2021.4.4 sacremoses-0.0.45 tokenizers-0.10.2
transformers-4.5.1 xmltodict-0.12.0
Singularity> python
>>> from transformers import TensorFlowBenchmark, TensorFlowBenchmarkArguments
>>> args = TensorFlowBenchmarkArguments( models=["bert-base-uncased"], batch_sizes=[512], sequence_lengths=[8, 32, 128] )
>>> benchmark = TensorFlowBenchmark(args)
>>> results = benchmark.run()
<… shortened …>
====================       INFERENCE - SPEED - RESULT       ====================
--------------------------------------------------------------------------------
          Model Name             Batch Size     Seq Length     Time in s
--------------------------------------------------------------------------------
      bert-base-uncased              512              8             0.024
      bert-base-uncased              512             32              0.1
      bert-base-uncased              512            128             0.406
-------------------------------------------------------------------------------- >>>
```

TensorFlow example

- Transformers benchmark
  - Python modules not in image – `transformers py3nvml`
  - **pip** – uses ~/.local/lib/ = outside the image (see docs of PYTHONUSERBASE)

```
$ qsub -I -q gpu -l select=1:ncpus=2:ngpus=1:mem=64gb:gpu_cap=cuda80:scratch_local=100gb -l walltime=8:00:00

zia5$ singularity shell --nv -B $SCRATCHDIR /cvmfs/singularity.metacentrum.cz/NGC/TensorFlow\:21.03-tf2-py3.SIF
Singularity> pip install transformers py3nvml
Successfully installed filelock-3.0.12 joblib-1.0.1 py3nvml-0.2.6 regex-2021.4.4 sacremoses-0.0.45 tokenizers-0.10.2
transformers-4.5.1 xmltodict-0.12.0
Singularity> python
>>> from transformers import TensorFlowBenchmark, TensorFlowBenchmarkArguments
>>> args = TensorFlowBenchmarkArguments( models=["bert-base-uncased"], batch_sizes=[512], sequence_lengths=[8, 32, 128] )
>>> benchmark = TensorFlowBenchmark(args)
>>> results = benchmark.run()
<… shortened …>
====================       INFERENCE - SPEED - RESULT       ====================
--------------------------------------------------------------------------------
          Model Name             Batch Size     Seq Length     Time in s
--------------------------------------------------------------------------------
      bert-base-uncased             512              8             0.024
      bert-base-uncased             512              32             0.1
      bert-base-uncased             512             128            0.406
-------------------------------------------------------------------------------- >>>
```

TensorFlow example

- Transformers benchmark - https://huggingface.co/transformers/benchmarks.html
  - Python modules not in image – `transformers py3nvml`
  - **pip** – uses ~/.local/lib/ = outside the image (see docs of PYTHONUSERBASE)

```
$ qsub -I -q gpu -l select=1:ncpus=2:ngpus=1:mem=64gb:gpu_cap=cuda80:scratch_local=100gb -l walltime=8:00:00

zia5$ singularity shell --nv -B $SCRATCHDIR /cvmfs/singularity.metacentrum.cz/NGC/TensorFlow\:21.03-tf2-py3.SIF
Singularity> pip install transformers py3nvml
Successfully installed filelock-3.0.12 joblib-1.0.1 py3nvml-0.2.6 regex-2021.4.4 sacremoses-0.0.45 tokenizers-0.10.2
transformers-4.5.1 xmltodict-0.12.0
Singularity> python
>>> from transformers import TensorFlowBenchmark, TensorFlowBenchmarkArguments
>>> args = TensorFlowBenchmarkArguments( models=["bert-base-uncased"], batch_sizes=[512], sequence_lengths=[8, 32, 128] )
>>> benchmark = TensorFlowBenchmark(args)
>>> results = benchmark.run()
<… shortened …>
====================       INFERENCE - SPEED - RESULT       ====================
--------------------------------------------------------------------------------
          Model Name             Batch Size     Seq Length     Time in s
--------------------------------------------------------------------------------
      bert-base-uncased             512              8              0.024
      bert-base-uncased             512              32              0.1
      bert-base-uncased             512             128             0.406
-------------------------------------------------------------------------------- >>>
```

NGC Summary

- Pros:
  - Easy to use
  - Optimized software and working
  - Excellent documentation
  - Docker images -> build own images derived form NGC
  - Repeatability

- Cons:
  - ?

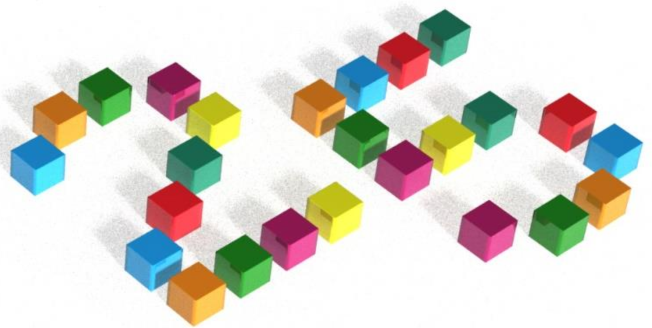
Notes for GPGPU in Metacentrum

- PBS queues `gpu@(meta|cerit)-pbs` have common environment
- PBS jobs start with `gpu_cap=cudaXX`, not `cl_adan=True`
- `CUDA_VISIBLE_DEVICES` – do not touch it ☺ and check foreign scripts.
- Use nvidia-smi to check GPU load during jobs tuning.
- The newest GPU is NOT the best for all jobs.
- Future of GPGPU
  - TensorCores, multi-GPU jobs
  - AMD – new GPU MI100 with tensor units, performace like A100
    - ROCm – equivalent of CUDA, similar principles

Notes for GPGPU in Metacentrum

- PBS queues `gpu@(meta|cerit)-pbs` have common environment
- PBS jobs start with `gpu_cap=cudaXX`, not `cl_adan=True`
- `CUDA_VISIBLE_DEVICES` – do not touch it ☺ and check foreign scripts.
- Use nvidia-smi to check GPU load during jobs tuning.
- The newest GPU is NOT the best for all jobs.
- Future of GPGPU
  - TensorCores, multi-GPU jobs
  - AMD – new GPU MI100 with tensor units, performance like A100
    - ROCm – equivalent of CUDA, similar principles

- Your questions ?



Thanks for your attention!

Děkuji za pozornost!

Jan Hoidekr, hoidekr@cesnet.cz