

GPGPU

- Informacje podstawowe
- Parametry kart GPGPU
- Zlecenie zadań
 - Dostęp do węzłów z GPGPU
 - Informacje ogólne
 - Zadania interaktywne
 - Zadania interaktywne MPI
 - Zadania wsadowe
 - Przykładowe skrypty dla systemu kolejkowego

Informacje podstawowe

Obliczenia GPGPU (*General-Purpose computing on Graphics Processing Units*) to wykorzystanie procesorów graficznych wspólnie z jednostką CPU do przyspieszenia obliczeń naukowych i inżynierskich. Infrastruktura PLGrid oferuje swoim użytkownikom dostęp do maszyny zawierającej karty GPU.

Lokalizacja	ACK CYFRONET AGH	
Nazwa systemu	Prometheus K40XL	Prometheus V100
Nazwa maszyny dostępowej	pro.cyfronet.pl	pro.cyfronet.pl
Port dostępowy	22	22
Liczba rdzeni obliczeniowych		
Liczba kart GPU	144	32
Kolejka	plgrid-gpu	plgrid-gpu-v100
Oprogramowanie	TeraChem, Gromacs, NAMD, GAMESS, TensorFlow, Keras, PyTorch	
Opis konfiguracji zasobów obliczeniowych	www.plgrid.pl/oferta/zasoby_obliczeniowe/opis_zasobow/HPC	
Opis systemów składowania danych	www.plgrid.pl/oferta/zasoby_obliczeniowe/opis_zasobow/storage	
Informacje rozszerzone	Dokumentacja KDM	
Kontakt	https://helpdesk.plgrid.pl	

Parametry kart GPGPU

Parametr	Wartość	
Dostępność	Prometheus (plgrid-gpu)	Prometheus (plgrid-gpu-v100)
Producent	NVIDIA	NVIDIA
Model	K40 XL	V100
Architektura	Kepler	Volta
Szyna	PCI-Express 3.0 16x	NVLink
Liczba rdzeni Tensor	-	640
Liczba rdzeni CUDA	2880	5120
Maksymalna częstotliwość	928 MHz	1290 MHz
Moc obliczeniowa (HP)	-	31.33 Tflops
Moc obliczeniowa (DP)	1,78 Tflops	7,834 Tflops
Moc obliczeniowa (SP)	5,34 Tflops	15,67 Tflops
Pojemność i typ pamięci	12 GB GDDR5	32 GB HBM2
Przepustowość pamięci	288 GB/s	900 GB/s

Zlecanie zadań

Dostęp do węzłów z GPGPU

Dla zadań korzystających w obliczeniach z kart GPGPU przeznaczona została specjalna partycja - **plgrid-gpu**. Aby móc przeprowadzać obliczenia z wykorzystaniem GPGPU na klastrze **Prometheus** konieczne jest złożenie wniosku o grant właściwy, który przeznaczony zostanie w całości wyłącznie na obliczenia z wykorzystaniem kart GPGPU. Grant taki nie powinien być używany do przeprowadzania obliczeń w partycjach innych niż **plgrid-gpu**. We wniosku o grant należy wyraźnie zaznaczyć, że wymagany jest dostęp do partycji **plgrid-gpu**. Zalecane jest także (ale nie jest to konieczne), aby grant taki posiadał w nazwie wyraz **gpu** (np. obliczeniagpu) - ułatwia to identyfikację grantów. Każdy wniosek o dostęp do partycji **plgrid-gpu** jest rozpatrywany indywidualnie przez dostawcę zasobów.

Dodatkowo dla obliczeń AI została udostępniona partycja **plgrid-gpu-v100** posiadająca karty GPGPU NVIDIA Tesla V100. Dostęp do nich realizowany jest podobnie, jak opisano powyżej dla partycji plgrid-gpu.

Informacje ogólne

Karty GPU w systemie kolejkowym SLURM są rodzajem tzw. generic resources (GRES), a ich identyfikatorem jest "gpu". Informację o tym na których węzłach/partycjach znajdują się karty GPU można otrzymać np. przy pomocy komendy sinfo:

```
sinfo -o '%P || %N || %G'
```

Zlecanie zadań odbywa się poprzez podanie opcji `--partition=plgrid-gpu --gres=gpu[:count]` systemu kolejkowego. W przypadku gdy nie ma podanej opcji count, system kolejkowy domyślnie alokuje jedną kartę na węźle obliczeniowym.

Po zleceniu zadania system kolejkowy automatycznie ustawia zmienną środowiskową `$CUDA_VISIBLE_DEVICES` oraz zezwala na dostęp do zaalokowanych do kart.

Zadania interaktywne

Przykładowo, gdy chcemy uruchomić zadanie interaktywne na jednym serwerze i zażądać 2 kart GPU:

```
srun -p plgrid-gpu -N 1 -n 24 -A <grant_id> --gres=gpu:2 --pty /bin/bash -l
```

Zadania interaktywne MPI

Uwaga! Wyjątkiem jest uruchamianie interaktywnych aplikacji MPI (np. w celach testów). Z powodu nieco innej obsługi kart GPU niż zwykłych procesorów przez system SLURM, uruchomienie zadania interaktywnego wymaga niestandardowej procedury.

Przykładowo, gdy chcemy uruchomić zadanie interaktywne na dwóch serwerach i zażądać 2 kart GPU na każdym z nich (łącznie 4 karty GPU) na czas 1 godziny:

```
salloc -p plgrid-gpu -N 2 --ntasks-per-node 24 -n 48 -A <grant_id> --gres=gpu:2 --time 1:00:00
```

Polecenie `salloc` zaalokowało nasze zadanie i zwróciło jego numer, przykładowo 1234.

Następnie uruchamiamy `srun` wymagając 0 kart GPU w zadaniu o numerze zwróconym przez `salloc`:

```
srun --jobid=1234 --gres=gpu:0 -0 --pty /bin/bash -l
```

Otrzymaliśmy dostęp do powłoki na jednym z węzłów, teraz po załadowaniu odpowiedniego modułu MPI można już uruchomić własną aplikację za pomocą `mpirun` lub `mpirexec`. Istotne jest aby podczas uruchamiania aplikacji nie przekazywać zmiennych środowiska, czyli w przypadku IntelMPI należy dodać parametr `-genvnone`. Aplikacja będzie miała dostęp do wszystkich kart GPU zaalokowanych w komendzie `salloc`, niezależnie od wymagania `gpu:0` użytego w `srun`.

Po zakończeniu testów należy usunąć alokację za pomocą:

```
scancel 1234
```

Zadania wsadowe

Dla skryptu wsadowego dodatkowe zmiany nie są potrzebne, przykładowo gdy żądamy po jednej karcie na węzeł obliczeniowy, wystarczy dopisać do skryptu:

```
#SBATCH --partition=plgrid-gpu
#SBATCH --gres=gpu
```

Uwaga: Wszelkie informacje na temat komend SLURMa można znaleźć w manualu, np.: `man sbatch`

Przykładowe skrypty dla systemu kolejkowego

TeraChem (zrównoleglenie na dwie karty graficzne)

```
#!/bin/bash
#SBATCH -N 1
#SBATCH --ntasks-per-node=1
#SBATCH -p plgrid-gpu
#SBATCH --gres=gpu:2
#SBATCH --time 1:00:00

# initializing proper environment for TeraChem
plgrid/apps/terachem/1.93

# actual job
$TERACHEMRUN ch.inp > ch.log
```

NAMD

```
#!/bin/bash
# NAMD GPPGU requires exactly one working node
#SBATCH -N 1
#SBATCH --ntasks-per-node=24
#SBATCH -p plgrid-gpu
#SBATCH --gres=gpu:2
#SBATCH --time 1:00:00

module load plgrid/apps/namd/2.14-mpi

cd $SLURM_SUBMIT_DIR

if [ ! -f stmv.tar.gz ]; then
    wget http://www.ks.uiuc.edu/Research/namd/utilities/stmv.tar.gz
fi
tar -xvf stmv.tar.gz -C $SCRATCHDIR

cd $SCRATCHDIR/stmv

sed -i 's/500/5000/g' stmv.namd
sed -i 's/\/usr\/tmp\/stmv-output\/\${env}\(SCRATCHDIR\)\/stmv-output/g' stmv.namd

namdrun stmv.namd
```