Tutorial 2: Cluster access and Parallel Computing

Informatik elective: GPU Computing

Pratik Nayak

In this session

- Cluster credentials and access
    - Navigating the cluster and check specifications
- Cluster performance check
- Estimate cluster performance: Peak FLOP/s, memory bandwidth, etc
- Clone the main exercise repository.
    - Solve exercises

[HANDS-ON] Login to cluster

- Login to the cluster:
  - Ensure you are logged into to eduVPN (you might face issues with eduroam)
  - `ssh -i <your_ssh_key> <username>@vm.ginkgo-project.de`
  - `ssh -i <your_ssh_key> <username>@10.152.225.226`
- `pwd` and check `/home/<username>` exists
- Does `nvidia-smi` show the available GPUs ?

[HANDS-ON] Clone the exercise framework repo

- We will use `gitlab.lrz.de` as the remote repository for the exercises.
- Create an account on `gitlab.lrz.de`
  - Login and update username
  - Tell me your username and I can add you to the group.
- Clone the repository:

```
git clone https://gitlab.lrz.de/2024ws-gpu-computing/exercises.git
```

[HANDS-ON] Run the hello world GPU example

- Build the code with CMake
- CUDA and CMake should already be available (`cmake --version` and `nvcc --version`).
- Regular build with CMake should link with CUDA and create an executable (See previous tutorial slides).
- Run the executable and check if you get output information on the current GPU.

[HANDS-ON] Arithmetic intensity

- Look at the example for the dot product, and implement the other operations:
    - axpy: `y = y + alpha*x`, where `y` and `x` are arrays and `alpha` is a scalar.
    - matvec: `x = A*b`, where `A` is a matrix and `b` is a vector.
    - matmul: `C = A*B`, where `A` and `B` are matrices
- Measure the performance in GFlops/s of these operations (with increasing input sizes) and plot a roofline on your local system, and also on the cluster.
- Are these operations memory bound/compute bound ?