Introduction to GPU Computing with CUDA

Dorothea vom Bruch

May 2019
5th Infieri Summer School
HUST, Wuhan, China
Why GPU computing?

- Moore’s law no longer valid
- Use many cores instead
- One option: GPUs

- Computing load in science growing rapidly, both for theory and experiments
- In many cases, GPUs are used to speed up computations
How to do it myself?

- Learn the basics of Nvidia’s API CUDA for programming GPUs
- Easiest entry point for GPU programming
- Very similar to C, C++ with some extensions

**In this lab**

- Introduction to CUDA
- GPU’s memory hierarchy
- How to parallelize a given problem
GPU memory

- How to access the memory
- What can it be used for?
- How fast is it?
- Which limitations do I have to consider?

Host → 11 GB Global Memory → 64 kB Constant Memory → 0 (kB) Texture Cache, L2 Cache

Block 0

- 48 kB Shared Memory
- Registers
- Thread 0
- Thread 1
- ...

Block 1

- 48 kB Shared Memory
- Registers
- Thread 0
- Thread 1
- ...

D. vom Bruch
Parallelization

- How to divide a problem into tasks that can be handled by the cores?
- What about synchronization?
- How to communicate between the GPU and the CPU?
Lab session

**First:** Introduction to CUDA programming model and syntax

**Second:** Hands-on:

- Hello World
- Vector addition
- Matrix multiplication
- Learn about:
  - Threads, blocks
  - Shared memory
  - Caching of data in registers
Pre-requisites

• Some experience with C / C++ programming
• No experience with CUDA
• Need a laptop to connect to the server where the GPUs for this lab are located
  → Please verify that ssh works for you

• Please note that this lab is only available during lab sessions 1, 3, 5, 7, 9