LEVELS OF PARALLELISM AND HIGH PERFORMANCE COMPUTING

IEEE IGARSS 2021 Tutorial on Scalable Machine Learning with High Performance and Cloud Computing

DR. - ING. GABRIELE CAVALARO
HIGH PRODUCTIVITY DATA PROCESSING RESEARCH GROUP
JÜLICH SUPERCOMPUTING CENTRE
WWW.GABRIELE-CAVALLARO.COM
IT ENVIRONMENTS

Key Priorities

**Embedded system**
- Power consumption and price

**Desktop Computing**
- Performance / price ration

**Servers**
- Maximum performance and maintainability reliability

Levels of Parallelism and HPC
HOW TO BECOME FASTER?

Split into instructions

Levels of Parallelism and HPC
HOW TO BECOME FASTER?
Three Alternative Ways

Work Harder
Clock Speed

Work Smarter
Optimization, Caching

Get Help
Parallelization

Levels of Parallelism and HPC
VON NEUMANN ARCHITECTURE

Machine Model

- Memory
- Output
- Input
- Bus
- Control Unit
- Central Unit
- Arithmetic Logic Unit
MOORE'S LAW

Levels of Parallelism and HPC

Wikipedia: Transistor count
THE END OF DENNARD SCALING
Why haven’t clock speeds increased, even though transistors have continued to shrink?

Levels of Parallelism and HPC
Karl Rupp, 40 Years of Microprocessor Trend Data
https://www.karlrupp.net/2015/06/40-years-of-microprocessor-trend-data/
WILL MOORE’S LAW END?
Paradigm Shift

“There’s no getting around the fact that we build these things out of atoms”

Gordon Moore
WHY SUPERCOMPUTER PERFORMANCE KEEP INCREASING?

Parallel computing is going mainstream

Levels of Parallelism and HPC

Rob Schreiber, High Performance Computing: Beyond Moore’s Law
https://www.youtube.com/watch?v=LOf5T5iXix4&t=258s
THE EASY TIMES HAVE GONE

Responsibility for better performance is on the software developers
PARALLEL COMPUTING

Levels of Parallelism and HPC
SINGLE INSTRUCTION MULTIPLE DATA (SIMD)
In-core parallelism

Parallel execution within a single CPU core

Operation with Vectors

Elements of the vector are sum in parallel
SIMULTANEOUS MULTITHREADING (SMT)

In-core parallelism

Split physical cores into virtual cores (threads)

Thread
CORE
Thread
CORE
CORE
CORE

Multiple threads with different tasks are executed simultaneously on one CPU core

Levels of Parallelism and HPC
GRAPHICS PROCESSING UNIT (GPU) VS CPU
Made of many simple Cores

Levels of Parallelism and HPC
NVIDIA AMPERE GPU ARCHITECTURE

Streaming Processors (SPs)

SP (CUDA core)

Compute elements:
16 32-bit integer point units
16 32-bit floating point units
8 64-bit floating point units

Ronny Krashinsky, et al., NVIDIA Ampere Architecture In-Depth
https://developer.nvidia.com/blog/nvidia-ampere-architecture-in-depth/

Levels of Parallelism and HPC
SINGLE INSTRUCTION MULTIPLE THREADS (SIMT)

In-processor parallelism - SIMT= SIMD + SMT

The WARPS are executed simultaneously by a SM

The threads of a WARP execute the same instruction (as SIMD)
SHARED-MEMORY ARCHITECTURE
Single computer
PARALLEL PROGRAMMING MODEL: OPENMP

Single Computer

The programmer works on the parallelism, leaving the data shuffling to the hardware system.
DISTRIBUTED-MEMORY ARCHITECTURE

Multiple Computers

Levels of Parallelism and HPC
PARALLEL PROGRAMMING MODEL: MPI

Multiple Computers

Message Passing Interface (MPI)

P1 P2 P3 P4 P5 Processes

Forces the programmer to consider the distribution of the data as a first-class concern
WHAT IS A SUPERCOMPUTER?

Mixture of shared-memory and distributed-memory architectures

Levels of Parallelism and HPC
WHAT IS NOT A SUPERCOMPUTER?

Computer cluster
Many computers bound together locally

Grid computing
Many clusters working as some kind of supercomputer

Cloud computing
Virtual machines in compute center(s)

High Throughput Computing
Many independent computations

Difference between Grid Computing and Cloud Computing
http://www.differencebetween.net/
TOP500 LIST
November 2020

Fugaku
442,010 Tflops/s

Summit
148,600 Tflop/s

1

Sierra
94,640 Tflop/s

2

3

TOP500

NVIDIA DGX SuperPOD
26,195 Gflops/watts

MN-3
26,039 Gflops/watts

1

2

JUWELS Booster Module
25,008 Gflops/watts

3

GREEN500

Levels of Parallelism and HPC
A RACE TOWARD EXASCALE COMPUTING

Levels of Parallelism and HPC
MODULAR SUPERCOMPUTING ARCHITECTURE (MSA)

Heterogeneous HPC clusters (modules) within a single system
REVIEW ON HARDWARE LEVELS OF PARALLELISM

Best performance is achieved with a combination of them!

- **SIMD**: In-core parallelism
- **SIMT**: In-processor parallelism
- **SMT**: Single Computer
  - Simultaneous Multithreading
  - Cross-core, Cross-socket
  - OpenMP, pthreads
- **MPI**: Multiple “Computers”
  - Tightly-coupled Supercomputing
- **MPI+MSA**: Multiple HPC Systems
  - Tightly-coupled Heterogeneous Hardware

Levels of Parallelism and HPC
ANATOMY OF A SUPERCOMPUTER

Levels of Parallelism and HPC
SUPERCOMPUTER USAGE MODEL
JUSUF
Jülich Support for Fenix

Partition with GPUs
61 NVIDIA V100 GPU with 16 GB Memory

https://apps.t2-juelich.de/jsc/hpc/jusuf/cluster/configuration.html

https://fenix-ii.eu/
PRE-PRACTICAL – STEP 1
Go to https://jupyter-jsc.fz-juelich.de/, login and add a new JupyterLab
STEP 2
Select the options and Start
STEP 3
Obtain the tutorial folder

- Open terminal

- Run this command and the tutorial folder training2118 will appear in the navigator on the left.
STEP 4
Create your own folder

- Run the following commands in the terminal to navigate to the tutorial folder “training2118” and create your own folder

```
cavallaro1@jsfc041:/p/project
[cavallaro1@jsfc041 ~]$ cd /p/project/training2118/
[cavallaro1@jsfc041 training2118]$ mkdir $USER
```

- You can check if the folder was created by looking at the navigator on the left
STEP 5
Download Jupyter notebook

- Go to https://www.gabriele-cavallaro.com/teaching/tutorial-igarss2021
- Download the Jupyter notebook "show_resources.ipynb" of Lecture 2
STEP 6
Upload the Jupyter notebook in your own folder
STEP 7
Run the Jupyter notebook

- Select the kernel PyDeepLearning-1.0
STEP 8
TERMINAL / Shell

```bash
[cavallaro1@jsfc041 ~]$ module unload nvidia-driver/.default
[cavallaro1@jsfc041 ~]$ srun nvidia-smi
Fri Jul  9 16:28:37 2021

NVIDIA-SMI 460.32.03 Driver Version: 460.32.03 CUDA Version: 11.2

<table>
<thead>
<tr>
<th>GPU Name</th>
<th>Persistence-M</th>
<th>Bus-ID</th>
<th>Disp.A</th>
<th>Memory-Usage</th>
<th>Volatile Uncorr. ECC</th>
<th>GPU-Util</th>
<th>Compute M.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 Tesla V100-PCIE...</td>
<td>On</td>
<td>00000000:01:00.0</td>
<td>Off</td>
<td>0%</td>
<td>16160MiB</td>
<td>0%</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Processes:
<table>
<thead>
<tr>
<th>GPU</th>
<th>GI</th>
<th>CI</th>
<th>PID</th>
<th>Type</th>
<th>Process name</th>
<th>GPU Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Usage</td>
</tr>
<tr>
<td>No running processes found</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
STEP 9
Access on the login node

- Starting point, in this node you do not have GPUs
TOMORROW PRACTICALS

Hands-on - Distributed Deep Learning

- Workflow on the batch system
- Use job scripts to execute algorithms with more nodes (i.e., > 1 GPUs)