

LEVELS OF PARALLELISM AND HIGH PERFORMANCE COMPUTING

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

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













IT ENVIRONMENTS





Desktop Computing







HOW TO BECOME FASTER?

Split into instructions



HOW TO BECOME FASTER?

Three Alternative Ways







VONNEUMANNARCHITECTURE





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?



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



Rob Schreiber, High Performance Computing: Beyond Moore's Law https://www.youtube.com/watch?v=L0f57fdxIn4&t=258s

THE EASY TIMES HAVE GONE

Responsibility for better performance is on the software developers



PARALLEL COMPUTING



SINGLE INSTRUCTION MULTIPLE DATA (SIMD) In-core parallelism





SIMULTANEOUS MULTITHREADING (SMT)

In-core parallelism



GRAPHICS PROCESSING UNIT (GPU) VS CPU

Made of many simple Cores



NVIDIA AMPERE GPU ARCHITECTURE

Streaming Multiprocessors (SMs)



NVIDIA Ampere Architecture (128 SMs)

Levels of Parallelism and HPC

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



NVIDIA AMPERE GPU ARCHITECTURE

Streaming Processors (SPs)



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

SP (CUDA core)												
	L0 Instruction Cache											
		War	p Sch	edule	r (32 t	hread	Vclk)	1				
-		Di	spatch	1 Unit	(32 th	read/o	cik)					
		Reg	ister	File ('	16,38	4 x 32	2-bit)					
INT32	INT32	FP32	FP32	FP	64							
INT32	INT32	FP32	FP32	FP	64							
INT32	INT32	FP32	FP32 FP32 FP64									
INT32	INT32	FP32	FP32	FP	64	TENSOR CORE						
INT32	INT32	FP32	FP32	FP	64	TENSOR CORE						
INT32	INT32	FP32	FP32	FP	64							
INT32	INT32 INT32 FP32 FP32 FP64											
INT32	INT32	FP32	FP32	FP	64							
LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	SFU				

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

SINGLE INSTRUCTION MULTIPLE THREADS (SIMT)

The WARPS are executed simultaneously by a SM

The threads of a WARP execute the same instruction (as SIMD)

In-processor parallelism - SIMT = SIMD + SMT



els of Parallelism and HPC

Streaming Multiprocessor (SM)

						L1 instru							
	-	LO Ir	nstruc	tion C	ache	_			LOI	nstruc	tion C	ache	_
	War	p Sch	edule	r (32.1	hread/clk1	_		Wan	o Sch	redule	r (32 t	hread/cik)	_
-	Di	spatch	h Unit	(32 th	read/cik)	-		Dis	pate	h Unit	(32 th	read/clk)	_
	Reg	ister	File (16,38	4 x 32-bit)			Regi	ister	File (16,38	4 x 32-bit)	
FP64	INT	INT	FP32	FP32			FP64	INT	INT	FP32	FP32		
FP64	INT	INT	FP32	FP32			FP64	INT	INT	FP32	FP32		
FP64	INT	INT	FP32	FP32			FP64	INT	INT	FP32	FP32		
FP64	INT	INT INT FP32		FP32	TENSOR	TENSOR	FP64	INT	INT	FP32	FP32	TENSOR	TENSOR
FP64	INT			FP32	CORE	CORE	FP64	INT	INT	FP32 FP32	CORE	CORE	
FP64	INT	INT	FP32	FP32			FP64	INT	INT	FP32	FP32		
FP64	INT	INT	FP32	FP32			FP64	INT	INT	FP32	FP32		
FP64	INT	INT	FP32	FP32			FP64	INT	INT	FP32	FP32		
ST ST	ST ST	SFU	ST ST	ST	ST	ST	ST	ST ST	SFU				
							J						
		L0 Ir	struc	tion C	ache				LO Ir	hstruc	tion C	ache	_
	War	LO Ir p Sch	istruc iedule	tion C r (32 t	ache hread/clk)			War	LO Ir p Sch	nstruc nedule	tion C f (32 t	ache hread/cik)	
	War	LO Ir p Sch spatcl	istruc iedule h Unit	tion C r (32 t (32 th	ache hread/clk) read/clk)			War	LO Ir p Sch patcl	nstruc nedule h Unit	tion C r (32 t (32 th	ache hread/clk) read/clk)	
	War Di Reg	LO Ir p Sch spatcl ister	edule Duit File ('	tion C r (32 t (32 th 16,384	ache hread/clk) read/clk) 4 x 32-bit)			War Dis Rogi	LO Ir p Sch patch ister	nstruc hedule h Unit File ('	lion C r (32 t (32 th 16,384	ache hread/clk) read/clk) 4 x 32-bit)	
FP64	War Di Rog	LO ir p Sch spatcl ister INT	edule Unit Filo (* FP32	tion C r (32 th (32 th 16,384 FP32	ache hread/clk) read/clk) 4 x 32-bit)		FP64	War Dis Rogi INT	LO Ir p Sch patcl ister INT	nstruc hedule File (' FP32	tion C r (32 t (32 th 16,38 FP32	ache hread/clk) read/clk) 4 x 32-bit)	
FP54 FP64	War Di Rog INT	LO Ir p Sch spatch ister INT	edule b Unit Filo (* FP32	tion C (32 th (32 th 16,384 FP32 FP32	ache hread/clk) read/clk) 4 x 32-bit)		FP84 FP84	Warj Dis Rogi INT INT	LO Ir p Sch patcl ister INT	nstruc h Unit File (' FP32	tion C r (32 t (32 th 16,38 FP32 FP32	ache hread/clk) read/clk) 4 x 32-bit)	
FP64 FP84 FP84	War Dis Rog INT INT INT	LO Ir p Sch spatch ister INT INT	edule Dunit Filo (* FP32 FP32	tion C (32 th (32 th 16,38/ FP32 FP32 FP32	ache hread/clk) read/clk) 4 x 32-bit)		FP84 FP84 FP84	War Dis Rogi INT INT INT	LO Ir p Sch patcl ister INT INT	nstruc nedule h Unit File (' FP32 FP32	tion C (32 th (32 th 16,38 FP32 FP32 FP32	ache hread/clk) read/clk) 4 x 32-bit)	
FP84 FP84 FP84 FP64	War Di Rog INT INT INT	LO Ir p Sch spatch ister INT INT INT	edule b Unit File (* FP32 FP32 FP32 FP32	tion C (32 th (32 th 16,38/ FP32 FP32 FP32 FP32	ache hread/clk) read/clk) 4 x 32-bit) TENSOR	TENSOR	FP64 FP64 FP64 FP64	War Dis Rogi INT INT INT INT	LO II p Sch ipatcl istor INT INT INT	File (FP32 FP32 FP32 FP32	tion C (32 th (32 th 16,384 FP32 FP32 FP32 FP32	ache hread/clk) read/clk) 4 x 32-bit) TENSOR	TENSOR
FP64 FP64 FP64 FP64	War Di Rog INT INT INT INT	LO Ir p Sch spatch istor INT INT INT INT	edule edule h Unit Filo (* FP32 FP32 FP32 FP32	tion C (32 th (32 th 16,38/ FP32 FP32 FP32 FP32	ache hread/clk) 4 x 32-bit) TENSOR CORE	TENSOR	FP64 FP64 FP64 FP64 FP64	War Dis Rogi INT INT INT INT	LO Ir p Sch patcl istor INT INT INT INT	File (FP32 FP32 FP32 FP32 FP32 FP32	Hom C (32 th (32 th 16,38 FP32 FP32 FP32 FP32	ache hreadicik) readicik) 4 x 32-bit) TENSOR CORE	TENSOR
FP64 FP64 FP64 FP64 FP64	War Di Rog INT INT INT INT INT	LO In p Sch spatch ister INT INT INT INT	edule edule File (' FP32 FP32 FP32 FP32 FP32	tion C (32 th (32 th 16,38- FP32 FP32 FP32 FP32 FP32	ache hread/clk) f x 32-bit) TENSOR CORE	TENSOR	FP64 FP64 FP64 FP64 FP64	War Dis Rogi INT INT INT INT INT	LO In o Sch opatol ister INT INT INT INT	hstruc nedule h Unit Filo (FP32 FP32 FP32 FP32 FP32	Hon C (32 th (32 th 16,38 FP32 FP32 FP32 FP32 FP32	ache hread/clk) fead/clk) 4 x 32-bit) TENSOR CORE	TENSOR
FP64 FP64 FP64 FP64 FP64 FP64 FP64	War Di Rog INT INT INT INT INT INT	LO In p Sch spatci ister INT INT INT INT INT	Istruc edule b Unit Filo (' FP32 FP32 FP32 FP32 FP32	tion C (32 th (32 th 16,384 FP32 FP32 FP32 FP32 FP32 FP32 FP32	ache hread/clk) 4 x 32-bit) TENSOR CORE	TENSOR	1994 1994 1994 1994 1994 1994	Ware Dis Rogi INT INT INT INT INT INT	LOB pSch spatcl istor INT INT INT INT INT	nedule nedule h Unit Filo (FP32 FP32 FP32 FP32 FP32	lion C (32 th (32 th 16,38 FP32 FP32 FP32 FP32 FP32 FP32	ache hread/cik) read/cik) 4 x 32-bit) 4 x 32-bit) TENSOR CORE	TENSOR
FP64 FP64 FP64 FP64 FP64 FP64 FP64 FP64	Wase Dis Rog INT INT INT INT INT INT	LO Ir p Sch spatch istor INT INT INT INT INT INT INT	edule edule funit File (FP32 FP32 FP32 FP32 FP32 FP32 FP32	tion C (32 th (32 th 16,38/ FP32 FP32 FP32 FP32 FP32 FP32 FP32	ache hread/clk) read/clk) 4 x 32-bit) TENSOR CORE	TENSOR	FP54 FP54 FP54 FP54 FP54 FP54 FP54 FP54	Warron	LO II p Sch ipatol ister INT INT INT INT INT INT INT	natruc nedule h Unit Filo (FP32 FP32 FP32 FP32 FP32 FP32	lion C (32 th (32 th 16,38 FP32 FP32 FP32 FP32 FP32 FP32 FP32	ache hread/cik) read/cik) 4 x 32-bit) TENSOR CORE	TENSOR
FP64 FP64 FP64 FP64 FP64 FP64 FP64 FP64	Week Dit INT INT INT INT INT INT INT	LO Ir p Sch spatch spatch istor INT INT INT INT INT INT LO ST	edule b Unit Filo (* FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	lion C (32 th (32 th 16,384 FP32 FP32 FP32 FP32 FP32 FP32 FP32 LD T	ache hread/clk) read/clk) 4 x 32-bit) TENSOR CORE	TENSOR CORE	PP64 PP64 PP64 PP64 PP64 PP64 PP64 PP64	Wary Dis Rogi INT INT INT INT INT INT INT INT	LO II patcl iptor ister INT INT INT INT INT INT INT	natzuc nedule h Unit Filo (FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	lion C (32 th (32 th 16,38 FP32 FP32 FP32 FP32 FP32 FP32 FP32 LD ST	ache hread/clk) 4 x 32-bit) TENSOR CORE	TENSOR CORE
FP64 FP94 FP94 FP64 FP64 FP64 FP64 LLV LLV ST LLV	Winner Die INT INT INT INT INT INT INT INT	LO Ir p Sch spatcl istor INT INT INT INT INT INT INT INT	edule s Unit Filo (* FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	tion C (32 th (32 th 16,38 FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	ache hread/cik) 4 x 32-bit) TENSOR CORE	TENSOR CORE SFU	РР44 РР44 РР44 РР44 РР44 РР44 РР44 РР4	Wary Dis Rogi INT INT INT INT INT INT INT INT INT INT	LO II Sch Ipatcl Istor INT INT INT INT INT INT INT INT INT	nstruc nedule h Unit FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	lion C (32 th (32 th 16,38 FP32 FP32 FP32 FP32 FP32 FP32 FP32 FP32	ache hread/clk) read/clk) 4 x 32-bit) TENSOR CORE	TENSOR CORE

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

SHARED-MEMORY ARCHITECTURE

Single computer



PARALLEL PROGRAMMING MODEL: OPENMP

Single Computer



DISTRIBUTED-MEMORY ARCHITECTURE Multiple Computers



PARALLEL PROGRAMMING MODEL: MPI

Multiple Computers



WHAT IS A SUPERCOMPUTER?

Mixture of shared-memory and distributed-memory architectures



WHAT IS NOT A SUPERCOMPUTER?



Difference between Grid Computing and Cloud Computing http://www.differencebetween.net/

TOP500 LIST November 2020



A RACE TOWARD EXASCALE COMPUTING



TOP500, Performance Development ttps://www.top500.org/statistics/perfdeve

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	SIMT	SMT	MPI	MPI+MSA
In-core parallelism	In-processor parallelism	Single Computer	Multiple "Computers"	Multiple HPC Systems
'	Many threads on many cores	Simultaneous Multithreading Cross-core, Cross-socket OpenMP, pthreads	Tightly-coupled Supercomputing	Tightly-coupled Heterogeneous Hardware

ANATOMY OF A SUPERCOMPUTER



SUPERCOMPUTER USAGE MODEL



JUSUF Jülich Support for Fenix



https://fenix-ri.eu/

Partition with GPUs

61 NVIDIA V100 GPU with 16 GB Memory

https://apps.fz-juelich.de/jsc/hps/jusuf/cluster/configuration.html

PRE-PRACTICAL-STEP1

Go to https://jupyter-jsc.fz-juelich.de/, login and add a new JupyterLab

NameSystemAccount/ImageProjectPartitionReservationResourcesAccount/Image	Actions	

STEP 2 Select the options and Start



Page 32

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/projec×

[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

🖿 / training2118 /								
Name	^	Last Modified						
cavallaro1		2 days ago						
dataset		2 days ago						
exercise_1		2 days ago						
exercise_2		2 days ago						
🖿 sedona3		2 days ago						

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

Lecture 2: Levels of Parallelism and High Performance Computing Get lecture Get notebook

STEP 6 Upload the Jupyter notebook in your own folder

	+ 83	±	G	♦⁺
	🖿 / training2118 / cavallaro	01/		☆
4	Name		La	st Modified
GILLAD				
0				



	🖿 / training2118 / cavallaro1	☆	
4	Name		Last Modified
Gitlab	show_resources.ipynb		seconds ago
U			

STEP 7 Run the Jupyter notebook

Select the kernel PyDeepLearning-1.0



<pre>b + & b & c & code & f & c & st Show the computing devices available with tensorflow [1]: from tensorflow.python.client import device_libb def.get_available_gpus(); local_device_protos = device_lib.list_local_devices() return [x.name for x in local_device_protos if x.device_type == 'GPU'] [2]: get_available_gpus() [2]: ['/device:GPU:0'] Print more details [3]: device_lib.list_local_devices() [3]: [name: "/device:CPU:0" device_type: "CPU" memory_limit: 26833456 locality { } incarnation: 17025335093315400299, name: "/device:GPU" memory_limit: 15090375584 locality { bus_lit: 4 numa_node: 3 links {</pre>	🛛 La	unc	her				\times		show <u>.</u>	_resource	s.ipynł	2	٠		
<pre>Show the computing devices available with tensorflow [1]: from tensorflow.python.client import device_lib def_get_available_gpus(): local_device_protes = device_lib.list_local_devices() return [x.name for x in local_device_protes if x.device_type == 'GPU'] [2]: get_available_gpus() [2]: ['/device:GPU:0'] Print more details [3]: device_lib.list_local_devices() [3]: [name: "/device:CPU:0" device_type: "CPU" memory_limit: 268435456 locality { hcarnation: 17025335093315400299, name: "/device:GPU:0" device_type: "CPU" memory_limit: 1509875584 locality { bus_id: 4 numm_node: 3 links { } // Intervalue (CPU:0) // CPU:0 // CPU:0</pre>	8	+	Ж		Ĉ	►		G	**	Code	~	Ø	0	0	git
<pre>[1]: from tensorflow.python.client import device_lib def_gst_available_gpus(): local_device_protos = device_lib.list_local_devices() return [x.name for x in local_device_protos if x.device_type == 'GPU'] [2]: get_available_gpus() [2]: ['/device:GPU:0'] Print more details [3]: device_lib.list_local_devices() [3]: [name: "/device:CPU:0" device_type: "CPU" memory_linit: 268435456 locality { } incarnation: 17025335093315400299, name: "/device:GPU:0" device_type: "GPU" memory_linit: 15509875584 locality { bus_fi: 4 numa_node: 3 links { } } }</pre>				Show	v the	com	putin	g dev	vices	available	with i	tenso	rflow		
<pre>def.get_available_gpus(): local_device_protos = device_lib.list_local_devices() return [x.name for x in local_device_protos if x.device_type == 'GPU'] [2]: get_available_gpus() [2]: ['/device:GPU:0'] Print more details [3]: device_lib.list_local_devices() [3]: [name: "/device:CPU:0" device_type: "CPU" memory_limit: 268435456 locality { } incarnation: 17025335093315400299, name: "/device:GPU:0" device_type: "CPU" memory_limit: 15509875584 locality { bus_id: 4 numa_node: 3 links { } // */// *///</pre>		[1]:	from	tens	sorfl	low.p	ytho	n.cl:	ient im	oort d	levice	e_lib)	
<pre>[2]: get_available_gpus() [2]: ['/device:GPU:0'] Print more details [3]: device_lib.list_local_devices() [3]: [name: "/device:CPU:0" device_type: "CPU" memory_limit: 268435456 locality { } incarnation: 17025335093315400299, name: "/device:GPU:0" device_type: "GPU" memory_limit: 1559937584 locality { bus_id: 4 numa_node: 3 links { } } }</pre>				<pre>def_get_available_gpus(): local_device_protos = device_lib.list_local_devices() return [x.name for x in local_device_protos if x.device_type == 'GPU']</pre>											
<pre>[2]: ['/device:GPU:0'] Print more details [3]: device_lib.list_local_devices() [3]: [name: "/device:CPU:0" device_type: "CPU" memory_limit: 268435456 locality { } incarnation: 17025335093315400299, name: "/device:GPU:0" device_type: "GPU" memory_limit: 15509875584 locality { bus_id: 4 numa_node: 3 links { } } </pre>		[2]:	get_	avail	lable	e_gpu	s()							
<pre>Print more details [3]: device_lib.list_local_devices() [3]: [name: "/device:CPU:0" device_type: "CPU" memory_limit: 268435456 locality { } incarnation: 17025335093315400299, name: "/device:GPU:0" device_type: "GPU" memory_limit: 15509875584 locality { bus_id: 4 numa_node: 3 links { } } }</pre>		[2]:	['/d	evice	e:GPU	J:0']								
<pre>[3]: device_lib.list_local_devices() [3]: [name: "/device:CPU:0" device_type: "CPU" memory_limit: 268435456 locality { } incarnation: 17025335093315400299, name: "/device:GPU:0" device_type: "GPU" memory_limit: 15509875584 locality { bus_id: 4 numa_node: 3 links { } } } }</pre>				Print	more	e det	ails								
<pre>[3]: [name: "/device:CPU:0" device_type: "CPU" memory_limit: 268435456 locality { } incarnation: 17025335093315400299, name: "/device:GPU:0" device_type: "GPU" memory_limit: 15509875584 locality { bus_id: 4 numa_node: 3 links { } }</pre>		[3]:	devi	ce_li	ib.li	st_l	ocal	_dev:	ices()					
} incarnation: 15490038752371625473 physical_device_desc: "device: 0, name: Tesla V100-PCIE-16GB, pci bus id: 0000:01:00.0, compute capability: 7.0"]		C	3]:	[name dev: memo loca } inca name dev: memo loca bi na loca } ; inca phy:	e: "/ ice_t ory_] ality arnat e: "/ ice_t ory_] ality us_ic uma_r inks arnat sica]	/devi type: limit /devi type: limit type: limit d: 4 foode: { tion: l_dev	170 170 170 170 170 170 170 170 170 170	PU:0 U" 8435 2533 PU:0 U" 5098 9003 desc	" 5093: " 75584 8752: : "de	31540029 4 3716254; evice: (99, 73	ae: T€	esla	V100	PCIE-16GB, pci bus id: 0000:01:00.0, compute capability: 7.0"]

STEP 8 TERMINAL / Shell



savallaro1@jsfc041:~ ×											
[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										
GPU Name Persistence-M Bus-Id Disp Fan Temp Perf Pwr:Usage/Cap Memory-Usa 	.A Volatile Uncorr. ECC ge GPU-Util Compute M. MIG M.										
0 Tesla V100-PCIE On 00000000:01:00.0 O N/A 27C P0 26W / 250W 0MiB / 16160M	ff 0 iB 0% Default N/A										
++											
Processes: GPU GI CI PID Type Process name ID ID	GPU Memory Usage										
No running processes found											
++ [cavallaro1@jsfc041 ~]\$											

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)

