Performance-portability and task parallelism for Lattice Field Theory

Bartosz Kostrzewa*, Stefan Krieg[†], Estela Suarez[‡]

*High Performance Computing & Analytics Lab [†]Helmholtz Zentrum für Strahlen und Kernphysik [‡]Institut für Informatik Rheinische Friedrich-Wilhelms-Universität Bonn [†]Juelich Supercomputing Center, Forschungszentrum Juelich

Colour meets Flavour – Innovative Algorithms Lab Kick-off meeting 2024.03.20



- Assume that sampling problems in lattice field theory can be tempered or overcome:
- critical slowing down
- ergodicity issues in the presence of multiple phases
- volume scaling of generative flow models
- multi-level sampling

- Assume that sampling problems in lattice field theory can be tempered or overcome:
- critical slowing down
- ergodicity issues in the presence of multiple phases
- volume scaling of generative flow models
- multi-level sampling
- What are the major technical challenges facing moonshot-type calculations?

- Assume that sampling problems in lattice field theory can be tempered or overcome:
- critical slowing down
- ergodicity issues in the presence of multiple phases
- volume scaling of generative flow models
- multi-level sampling
- What are the major technical challenges facing moonshot-type calculations?
- HPC heterogeneity: need write-once run-anywhere (as far as possible).

- Assume that sampling problems in lattice field theory can be tempered or overcome:
- critical slowing down
- ergodicity issues in the presence of multiple phases
- volume scaling of generative flow models
- multi-level sampling
- What are the major technical challenges facing moonshot-type calculations?
- HPC heterogeneity: need write-once run-anywhere (as far as possible).
- Strong scalability: most efficient algorithms don't scale well (maybe they don't need to).

- Assume that sampling problems in lattice field theory can be tempered or overcome:
- critical slowing down
- ergodicity issues in the presence of multiple phases
- volume scaling of generative flow models
- multi-level sampling
- What are the major technical challenges facing moonshot-type calculations?
- HPC heterogeneity: need write-once run-anywhere (as far as possible).
- Strong scalability: most efficient algorithms don't scale well (maybe they don't need to).
- Complex observables lead to hard-to-debug nested loops with horrible inter-dependencies and low re-use potential.

- Assume that sampling problems in lattice field theory can be tempered or overcome:
- critical slowing down
- ergodicity issues in the presence of multiple phases
- volume scaling of generative flow models
- multi-level sampling
- What are the major technical challenges facing moonshot-type calculations?
- HPC heterogeneity: need write-once run-anywhere (as far as possible).
- Strong scalability: most efficient algorithms don't scale well (maybe they don't need to).
- Complex observables lead to hard-to-debug nested loops with horrible inter-dependencies and low re-use potential.
- Lack of task parallelism: lots of untapped resources which are inaccessible with current LFT frameworks.

- Assume that sampling problems in lattice field theory can be tempered or overcome:
- critical slowing down
- ergodicity issues in the presence of multiple phases
- volume scaling of generative flow models
- multi-level sampling
- What are the major technical challenges facing moonshot-type calculations?
- HPC heterogeneity: need write-once run-anywhere (as far as possible).
- Strong scalability: most efficient algorithms don't scale well (maybe they don't need to).
- Complex observables lead to hard-to-debug nested loops with horrible inter-dependencies and low re-use potential.
- Lack of task parallelism: lots of untapped resources which are inaccessible with current LFT frameworks.
- Storage: we cannot store millions of gauge configurations \rightarrow might need on-the-fly calculations of everything.

- Assume that sampling problems in lattice field theory can be tempered or overcome:
- critical slowing down
- ergodicity issues in the presence of multiple phases
- volume scaling of generative flow models
- multi-level sampling
- What are the major technical challenges facing moonshot-type calculations?
- HPC heterogeneity: need write-once run-anywhere (as far as possible).
- Strong scalability: most efficient algorithms don't scale well (maybe they don't need to).
- Complex observables lead to hard-to-debug nested loops with horrible inter-dependencies and low re-use potential.
- **Lack of task parallelism**: lots of untapped resources which are inaccessible with current LFT frameworks.
- Storage: we cannot store millions of gauge configurations \rightarrow might need on-the-fly calculations of everything.
- Programmer productivity: need high-performance backends with easy to use frontends for students (and for ourselves).

- Assume that sampling problems in lattice field theory can be tempered or overcome:
- critical slowing down
- ergodicity issues in the presence of multiple phases
- volume scaling of generative flow models
- multi-level sampling
- What are the major technical challenges facing moonshot-type calculations?
- HPC heterogeneity: need write-once run-anywhere (as far as possible).
- Strong scalability: most efficient algorithms don't scale well (maybe they don't need to).
- Complex observables lead to hard-to-debug nested loops with horrible inter-dependencies and low re-use potential.
- Lack of task parallelism: lots of untapped resources which are inaccessible with current LFT frameworks.
- Storage: we cannot store millions of gauge configurations \rightarrow might need on-the-fly calculations of everything.
- Programmer productivity: need high-performance backends with easy to use frontends for students (and for ourselves).
- Physics students: not enough training in software engineering / HPC / modern C++ (for this kind of work)

HPC heterogeneity

(NUMERIQS)

Need to target different architectures



- Situation bound to get worse with future specialized hardware & Modular Supercomputing Architecture
- multi-purpose chiplet designs (EPI)
- in-memory computing
- neuromorphic devices
- tensor & stencil accelerators, large FPGAs

HPC heterogeneity

(NUMERIQS)

Need to target different architectures



- Situation bound to get worse with future specialized hardware & Modular Supercomputing Architecture
- multi-purpose chiplet designs (EPI)
- in-memory computing
- neuromorphic devices
- tensor & stencil accelerators, large FPGAs

xisting libraries w	vith some perform	ance-portability
chroma Public Contributors 13 (7) (7) (7) (7) (7) (7) (7) (7) (7) (7) (7) (7) (7)	5 quda Public Contributors 42 ☆ 유수 (응) 수 같은 42 구가 분용 ① 주 수 (야) 유수 수용 (응)	# Grid Pater Contributors # # @ 22 # # @ # @ 22 # # @ @ # @ 22 # # @ @ @ # @ 22 # # # @
Languages • C++ 96.3% • Perl 1.5% • Makefile 0.5% • Shell 0.6% • M4 0.4% • C 0.3%	+ 28 contributors Languages C++ 68.2% Cuda 25.0% C C3.6% CMake 2.0% Python 0.7% Shell 0.3% Other 0.2%	+ 27 contributors Languages C++ 92.9% C++ 92.9\% C++ 92.9

HPC heterogeneity

(NUMERIQS)

Need to target different architectures



- Situation bound to get worse with future specialized hardware & Modular Supercomputing Architecture
- multi-purpose chiplet designs (EPI)
- in-memory computing
- neuromorphic devices
- tensor & stencil accelerators, large FPGAs

Existing	libraries	with	some	performance-	portability

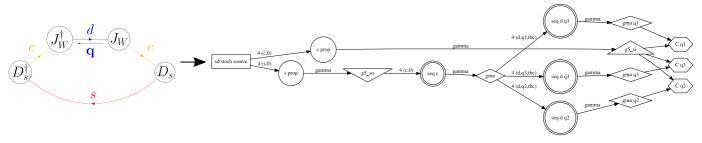
Contributors ¹³ S S S S S S S S S S S S S S S S S		
Languages	+ 28 contributors	Languages
Makefile 0.9% Shell 0.6% M4 0.4% C 0.3%	C++ 68.2% Cuda 25.0% C 3.6% CMake 2.0% Python 0.7% Shell 0.3% Other 0.2%	C++ 92.9% C 3.4% Mathematica 0.9% Shell 0.7% Roff 0.3%

Future directions explored in NuMeriQS B02

LQFT base layer: abstractions for memory space, memory access patterns and execution space, parallel patterns, asynchronicity via futures, etc.

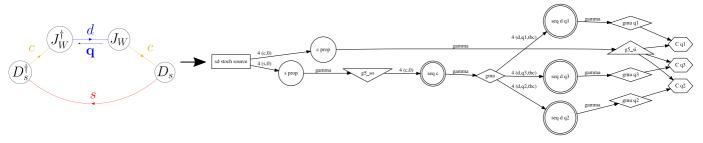
- std::execution
- std::mdspan
- std::async, std::future
- , 【 kokkos

- Simple observable: two current insertions with momentum transfer between meson initial and final states
- lots of unused parallelism at the inversion / contraction stage (different flavours, different momenta, different Dirac structures)



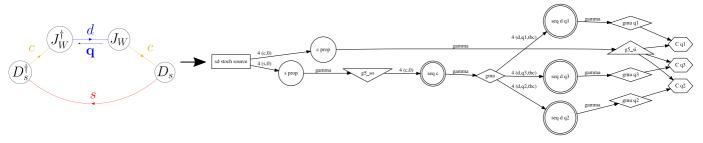
• How to exploit this without writing really complicated nested loops with tons of conditionals?

- Simple observable: two current insertions with momentum transfer between meson initial and final states
- lots of unused parallelism at the inversion / contraction stage (different flavours, different momenta, different Dirac structures)



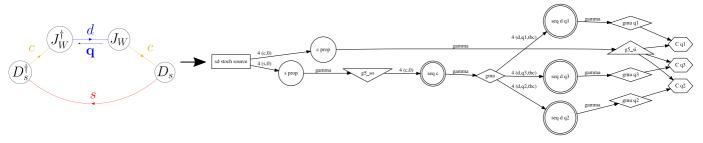
- How to exploit this without writing really complicated nested loops with tons of conditionals?
- Collections of observables as forests of directed acyclic graphs

- Simple observable: two current insertions with momentum transfer between meson initial and final states
- lots of unused parallelism at the inversion / contraction stage (different flavours, different momenta, different Dirac structures)



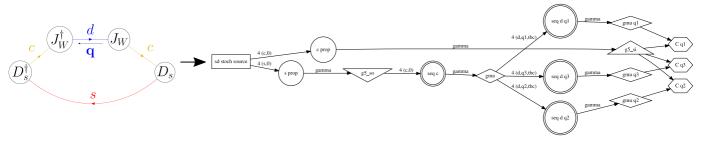
- How to exploit this without writing really complicated nested loops with tons of conditionals?
- ► Collections of observables as forests of directed acyclic graphs
- Edges carry computational cost

- Simple observable: two current insertions with momentum transfer between meson initial and final states
- lots of unused parallelism at the inversion / contraction stage (different flavours, different momenta, different Dirac structures)



- How to exploit this without writing really complicated nested loops with tons of conditionals?
- ► Collections of observables as forests of directed acyclic graphs
- Edges carry computational cost
- Vertices (may) carry computational and memory cost

- Simple observable: two current insertions with momentum transfer between meson initial and final states
- lots of unused parallelism at the inversion / contraction stage (different flavours, different momenta, different Dirac structures)



- How to exploit this without writing really complicated nested loops with tons of conditionals?
- Collections of observables as forests of directed acyclic graphs
- Edges carry computational cost
- Vertices (may) carry computational and memory cost
- ⇒ Optimise calculation under machine constraints
 - > all dependencies taken care of automatically, no more nested loops, just descend down the graph hierarchy
 - can trade memory or storage for computation or vice-versa

Task parallelism 〈NUMERIQS〉

					Version:			
					Version:			
						535.104.12		
			wr:Usag	nce-M e/Cap	Bus-Id 	Disp.A Memory-Usage		Uncorr. ECC Compute M. MIG M.
IVIDIA 52C	A100-9 P0					0:03:00.0 Off iB / 40960MiB		0 Default Disabled
IVIDIA 52C	A100-5 P0			0n 400W				0 Default Disabled
IVIDIA 51C	A100-9 P0			On 400W				0 Default Disabled
IVIDIA 50C	A100-9 P0		150W /	0n 400W				0 Default Disabled
GI ID								GPU Memory Usage
		18516 18518 18515 18515 18517		66	93758 - quda 93758 - quda	a-develop-dfei a-develop-dfei	f2c0/hmc_tm f2c0/hmc_tm	5700MiB 5700MiB 5700MiB 5700MiB 5700MiB
	52C VIDIA 52C VIDIA 51C VIDIA 50C Ses: GI ID N/A N/A N/A	52C P0 VIDIA A100-5 52C P0 VIDIA A100-5 51C P0 VIDIA A100-5 50C P0 Ses: GI CI ID ID N/A N/A N/A N/A N/A N/A N/A	VIDIA A100-SXM4-40GB 52C P0 VIDIA A100-SXM4-40GB 51C P0 VIDIA A100-SXM4-40GB 50C P0 Ses: GI CI PID ID ID ID ID N/A N/A 18516 N/A N/A 18515	52C P0 244W / VIDIA A100-SXM4-406B 264W / 52C P0 264W / VIDIA A100-SXM4-406B 166W / 51C P0 166W / VIDIA A100-SXM4-406B 156W / S0C P0 150W / Ses: GI CI PID JD ID ID Type N/A N/A 18516 C N/A N/A 18515 C	S2C P0 244W / 400W VIDIA A100-SXM4-406B 0n 52C P0 264W / 400W VIDIA A100-SXM4-406B 0n 51C P0 166W / 400W VIDIA A100-SXM4-406B 0n 50C P0 150W / 400W VIDIA A100-SXM4-406B 0n 50C P0 150W / 400W Sess: GI CI GI CI PID N/A N/A 18516 N/A N/A 18515 N/A N/A 18515	52C P0 244W / 400W 5884M VIDIA A100-SXM4-406B 264W / 400W 00000000 52C P0 264W / 400W 00000000 51C P0 166W / 400W 00000000 51C P0 166W / 400W 00000000 50C P0 150W / 400W 00000000 50C P0 150W / 400W 00000000 5884M 5884M 5884M 5884M VIDIA A100-SXM4-406B 00 00000000 5884M Sess: GI CI P0 5884M 5884M VIDIA A100-SXM4-406B 00 0 00000000 5884M Sess: GI CI P0 5884M 5884M 5884M VIDIA A100-SXM4-406B D0 0 00000000 5884M Sess: GI CI P0 5884M 5884M VIDIA A100-SXM4-406B D0 0 00000000 5884M Sess: GI CI P0 5884M 5884M 5884M	52C P0 244W / 400W 5884M1B / 40960M1B VIDIA A100-SXM4-406B On 00000000:44:00.0 Off 52C P0 264W / 400W 00000000:44:00.0 Off VIDIA A100-SXM4-406B On 00000000:84:00.0 Off VIDIA A100-SXM4-406B On 00000000:84:00.0 Off 50C P0 166W / 400W 00000000:C4:00.0 Off 50C P0 150W / 400W 00000000:C4:00.0 Off 50C P0 150W / 400W 00000000:C4:00.0 Off 50C P0 150W / 400W 00000000:C4:00.0 Off 5884M1B / 40960M1B 5884M1B / 40960M1B Ses: GI CI PID Type Process name 10 N/A N/A 18516 C N/A N/A 18515 C N/A N/A 18515 C	52C P0 244W / 400W I 5884M1B / 40960M1B 91% VIDIA A100-SXM4-40GB C64W / 400W I 00000000:44:00.0 Off 90% VIDIA A100-SXM4-40GB C64W / 400W I 00000000:84:00.0 Off 90% VIDIA A100-SXM4-40GB On I 00000000:84:00.0 Off 90% VIDIA A100-SXM4-40GB On I 00000000:84:00.0 Off 90% VIDIA A100-SXM4-40GB On I 00000000:C4:00.0 Off 90% Soc P0 I50W / 400W I 00000000:C4:00.0 Off 90% Ses: GI CI PID Type Process name ID ID Type Process name 90% N/A N/A 18516 C 6603758-quda-develop-dfef2co/hmc_tm N/A N/A 18515 C 6603758-quda-develop-dfef2co/hmc_tm

• Lots of free resources when running at strong-scaling limit.

ery 2	.0s: n	vidia-s	smi		u1003	239@mel2170:~91x38 mel2176): Tue Mar 1	9 15:34:59 2
	Name Temp			ersisten wr:Usage		Bus-Id Disp.A Memory-Usage		Uncorr. ECC Compute M. MIG M.
	====== NVIDIA 52C	====== A100-5 P0		244W /	0n 400W	00000000:03:00.0 Off 5884MiB / 40960MiB		0 Default Disabled
	NVIDIA 52C	A100-S P0			0n 400W	00000000:44:00.0 Off 5884MiB / 40960MiB		0 Default Disabled
	NVIDIA 51C	A100-S P0			0n 400W	00000000:84:00.0 Off 5884MiB / 40960MiB	 90% 	0 Default Disabled
	NVIDIA 50C	A100-S P0		150W /	0n 400W	000000000:C4:00.0 Off 5884MiB / 40960MiB	90%	0 Default Disabled
Proce GPU	GI GI ID							GPU Memory Usage
			18516 18518 18515 18515 18517		660	03758-quda-develop-dfef 03758-quda-develop-dfef 03758-quda-develop-dfef 03758-quda-develop-dfef 03758-quda-develop-dfef	2c0/hmc_tm 2c0/hmc_tm	======================================

- Lots of free resources when running at strong-scaling limit.
- GPU and CPU idle at different times.
- Wouldn't it be neat to fully use the CPU on a Grace-Hopper node?

2.0s: n	vidia-s	mi		u100	239@mel2170:~ 91x38 mel2170	: Tue Mar 1	9 15:34:59 2
	535.104						
Name Temp					Bus-Id Disp.A Memory-Usage 	Volatile GPU-Util 	Uncorr. ECC Compute M. MIG M.
NVIDIA 52C	======= A100-S P0	======== XM4-40GB	244W /	====== On 400W	00000000:03:00.0 Off 5884MiB / 40960MiB		0 Default Disabled
NVIDIA 52C	A100-S P0		264W /	0n 400W	00000000:44:00.0 Off 5884MiB / 40960MiB		0 Default Disabled
NVIDIA 51C	A100-S P0			0n 400W	00000000:84:00.0 Off 5884MiB / 40960MiB		0 Default Disabled
NVIDIA 50C	A100-S P0		150W /	0n 400W	00000000:C4:00.0 Off 5884MiB / 40960MiB		0 Default Disabled
GI ID	CI ID	PID	Туре	Proce	ss name		GPU Memory Usage
		======= 18516 18518 18515 18517		66 66	93758-quda-develop-dfef 93758-quda-develop-dfef	2c0/hmc_tm 2c0/hmc_tm	5700MiB 5700MiB 5700MiB 5700MiB 5700MiB
	A 19 15 CA-SMI Name Temp NVIDIA 52C NVIDIA 54C 74C 74C 74C 74C 74C 74C 74C 7	19 15:34:59 (A-SMI 535.104 Name Temp Perf NVIDIA A100-S 52C P0 NVIDIA A100-S 52C P0 NVIDIA A100-S 52C P0 NVIDIA A100-S 51C P0 NVIDIA A100-S 50C P0 SOC P0 SSESS: GI GI CI ID ID N/A N/A N/A N/A	Temp Perf P NVIDIA A100-SXM4-406B S 52C P0 SXM4-406B NVIDIA A100-SXM4-406B S NVIDIA A100-SXM4-406B S SSES: P0 S GI CI PID NA N/A 18516 N/A N/A 18515	19 15:34:59 2024 (A-SMI 535.104.12 Name Persiste Temp Perf Pwr:Usag NVIDIA A100-SXM4-40GB 244W / NVIDIA A100-SXM4-40GB 264W / S2C P0 264W / NVIDIA A100-SXM4-40GB 264W / NVIDIA A100-SXM4-40GB 166W / S1C P0 166W / NVIDIA A100-SXM4-40GB 150W / S5C P0 150W / S0C P0 150W / S9Sess: GI CI PID ID ID Type N/A N/A 18518 C N/A N/A 18515 C	2.0s: nvldia-smi 19 15:34:59 2024 (A-SMI 535.104.12 Driver Name Persistence-M Temp Perf Pwr:Usage/Cap NVIDIA A100-SXM4-406B On 52C P0 NVIDIA A100-SXM4-406B On 52C P0 NVIDIA A100-SXM4-406B On 52C P0 NVIDIA A100-SXM4-406B On 51C P0 NVIDIA A100-SXM4-406B On 52C P0 NVIDIA A100-SXM4-406B On 51C P0 NVIDIA A100-SXM4-406B On 56C P0 Seses: GI GI CI PID Type Proce: ID N/A N/A N/A N/A A18516 C C 664	2.0s: nvidia-smi mel2170 19 15:34:59 2024 1 KA-SMI 535.104.12 Driver Version: 535.104.12 Name Persistence-M Bus-Id Disp.A Temp Perf Pwr:Usage/Cap Memory-Usage NVIDIA A100-SXM4-40GB 0 0 00000000:03:00.0 0ff 52C P0 244W / 400W 1 0 NVIDIA A100-SXM4-40GB 0 0 00000000:44:00.0 0ff 52C P0 264W / 400W 1 00000000:44:00.0 0ff 52C P0 264W / 400W 1 00000000:44:00.0 0ff 52C P0 264W / 400W 1 00000000:44:00.0 0ff 51C P0 166W / 400W 1 00000000:64:00.0 0ff 51C P0 150W / 400W 1 00000000:C4:00.0 0ff 50C P0 150W / 400W 1 00000000:C4:00.0 0ff 5884M1B 40960M1B 1 00000000:C4:00.0 0ff 5884M1B / 40960M1B VIDIA A100-SXM4-40GB 0 0 00000000:C4:00.0 0ff 5884M1B / 40960M1B SSSES: GI CI	2.05: nvidia-smi mel2170: Tue Mar 1 19 15:34:59 2024 KA-SMI 535.104.12 Driver Version: 535.104.12 CUDA Version Name Persistence-M Bus-Id Disp.A Volatile Temp Perf Pwr:Usage/Cap Memory-Usage GPU-Util NVIDIA A100-SXM4-40GB On 00000000:03:00.0 0ff 91% S2C P0 244W / 400W 00000000:44:00.0 0ff 91% NVIDIA A100-SXM4-40GB On 00000000:44:00.0 0ff 90% NVIDIA A100-SXM4-40GB On 00000000:44:00.0 0ff 90% S1C P0 166W / 400W 00000000:24:00.0 0ff 90% NVIDIA A100-SXM4-40GB On 000000000:24:00.0 0ff 90% S5c P0 150W / 400W 00000000:24:00.0 0ff 90% S884M1B / 40960M1B 90% 90% 90% 90% Ssses: GI CI PID Type Process name ID ID Type Process name 90% 90% N/A N/A 18518 C 6603758-quda-develop-dfef2co/hmc_tm <t< td=""></t<>

- Lots of free resources when running at strong-scaling limit.
- GPU and CPU idle at different times.
- Wouldn't it be neat to fully use the CPU on a Grace-Hopper node?
- Would like to exploit this to run calculations in idle gaps, additional GPU streams.

ery 2	2.0s: n	vidia-s	mi		u100	239@mel2170:~ 9		0: Tue Mar :	19 15:34:59 2
		535.104							
GPU Fan	Name Temp			ersiste wr:Usag		Bus-Id	Disp.A Memory-Usage	Volatile GPU-Util 	Uncorr. ECC Compute M. MIG M.
	NVIDIA 52C	======= A100-S P0		244W /	0n 400W		:03:00.0 Off B / 40960MiB		0 Default Disabled
	NVIDIA 52C	A100-S P0		264W /	0n 7 400W		:44:00.0 Off B / 40960MiB		0 Default Disabled
	NVIDIA 51C	A100-S P0			On 400W		:84:00.0 Off B / 40960MiB		0 Default Disabled
	NVIDIA 50C	A100-S P0		150W /	On 400W		:C4:00.0 Off B / 40960MiB		0 Default Disabled
GPU	GI GI GI								GPU Memory Usage
			18516 18518 18515 18515 18517		66 66	93758 - quda 93758 - quda	-develop-dfe -develop-dfe -develop-dfe -develop-dfe	f2c0/hmc_tm f2c0/hmc_tm	5700MiB 5700MiB 5700MiB 5700MiB 5700MiB

- Lots of free resources when running at strong-scaling limit.
- GPU and CPU idle at different times.
- Wouldn't it be neat to fully use the CPU on a Grace-Hopper node?
- Would like to exploit this to run calculations in idle gaps, additional GPU streams.
- \Rightarrow Need suitable programming model to express task parallelism, work stealing, etc.

ery 2	2.0s: n	vidia-s	mi		u100	239@mel2170:~91x38 mel	2170: Tue Mar	19 15:34:59 20
		535.104						
GPU Fan	Name Temp			ersiste wr:Usag		Bus-Id Dis Memory-Us 		Uncorr. ECC Compute M. MIG M.
	======= NVIDIA 52C	====== A100-S P0		244W /	 On 400W	00000000:03:00.0 5884MiB / 40960		0 Default Disabled
	NVIDIA 52C	A100-S P0			0n 400W	00000000:44:00.0 5884MiB / 40960		0 Default Disabled
	NVIDIA 51C	A100-S P0			On 400W	00000000:84:00.0 5884MiB / 40960		0 Default Disabled
	NVIDIA 50C	A100-S P0		150W /	0n 400W	000000000:C4:00.0 5884MiB / 40960		0 Default Disabled
Proce GPU	esses: GI ID							GPU Memory Usage
			======= 18516 18518 18515 18517		66	======================================	dfef2c0/hmc_tm dfef2c0/hmc_tm	5700MiB 5700MiB

- Lots of free resources when running at strong-scaling limit.
- GPU and CPU idle at different times.
- Wouldn't it be neat to fully use the CPU on a Grace-Hopper node?
- Would like to exploit this to run calculations in idle gaps, additional GPU streams.
- \Rightarrow Need suitable programming model to express task parallelism, work stealing, etc.
 - ! Must be rather coarse-grained, existing asynchronous many task systems (likely) not suitable.

Summary (NUMERIQS)

- Assumption: sampling problems in lattice QFT will be tempered sufficiently
- produce millions of large volume gauge configurations
- ▶ impossible to store, need to calculate (almost) everything on the fly
- \Rightarrow Need: task parallelism, DAG representation of observables and dependencies, performance-portable kernels

Issue	Performance-portability	Task parallelism	$\textbf{loops} \rightarrow \textbf{DAGs}$	Productivity layer
HPC heterogeneity	 ✓ 	-	_	-
Strong scaling	-	V	V	-
Wasted resources		V	V	-
Complex observables	-	V	V	maybe
Storage	-	V	 Image: A set of the set of the	-
use ML frameworks	-	-	-	 ✓
Overwhelmed students	*	*	*	maybe