VASP Performance on HPE Cray EX Based on NVIDIA A100 GPUs and AMD Milan CPUs

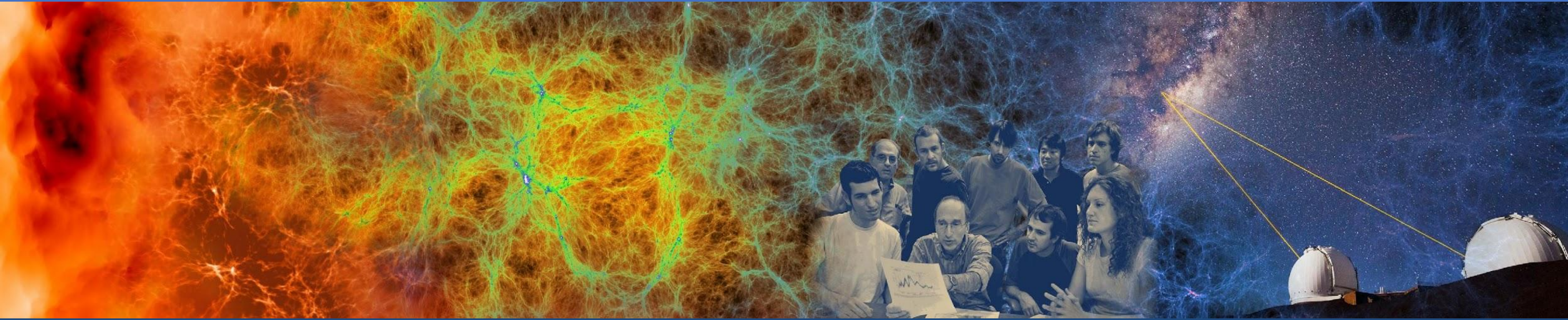
Zhengji Zhao[1)], Brian Austin[1)], Stefan Maintz[2)], Martijn Marsman[3)]
[1)] Lawrence Berkeley National Laboratory
[2)] NVIDIA, Inc
[3)] VASP Software GmbH & University of Vienna

May 11, 2023
Cray User Group Meeting 2023,
Helsinki, Finnland

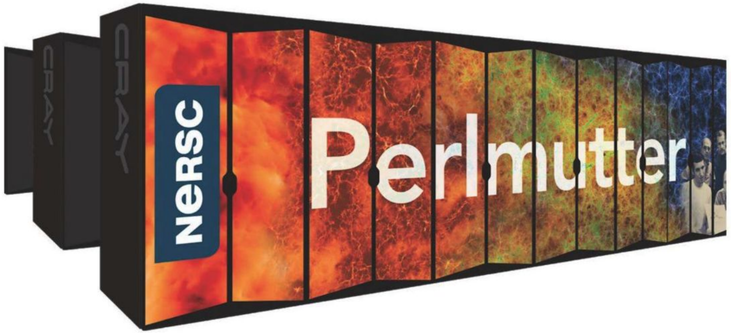
Outline

- Motivation
- System Configuration and Experimental Setup
- VASP Performance on Perlmutter GPUs and CPUs
- Energy Efficiency
- Summary

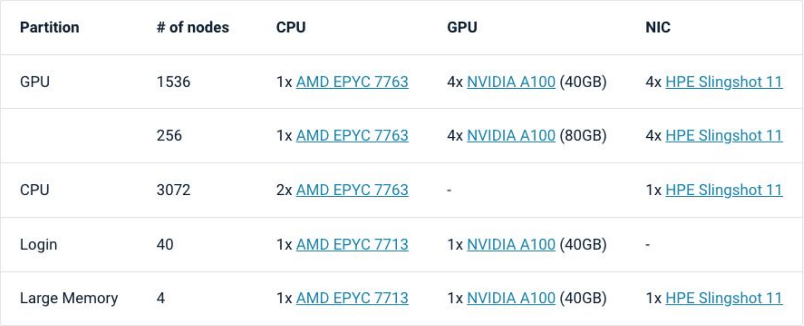
Motivation



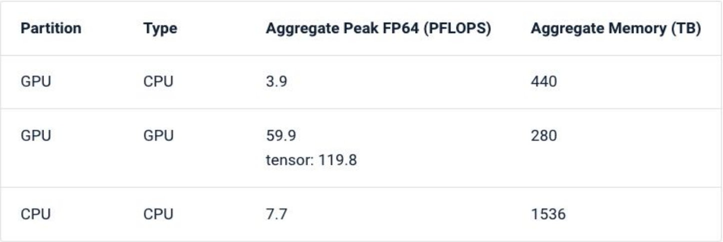
Background



Perlmutter System Specification and Performance



Partition	# of nodes	CPU	GPU	NIC
GPU	1536	1x AMD EPYC 7763	4x NVIDIA A100 (40GB)	4x HPE Slingshot 11
	256	1x AMD EPYC 7763	4x NVIDIA A100 (80GB)	4x HPE Slingshot 11
CPU	3072	2x AMD EPYC 7763	-	1x HPE Slingshot 11
Login	40	1x AMD EPYC 7713	1x NVIDIA A100 (40GB)	-
Large Memory	4	1x AMD EPYC 7713	1x NVIDIA A100 (40GB)	1x HPE Slingshot 11

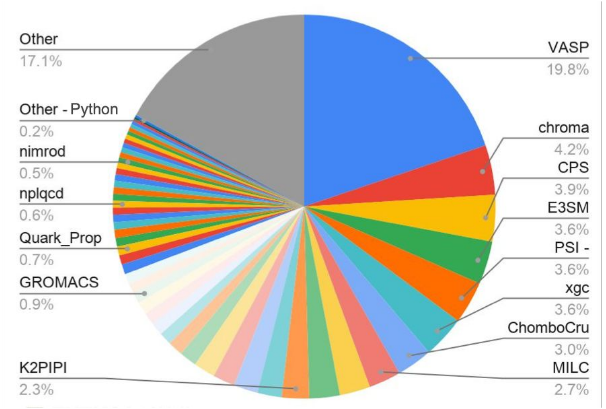


Partition	Type	Aggregate Peak FP64 (PFLOPS)	Aggregate Memory (TB)
GPU	CPU	3.9	440
GPU	GPU	59.9 tensor: 119.8	280
CPU	CPU	7.7	1536

- Perlmutter, NERSC's new pre-exascale system, has entered production!
- Users are transitioning to Perlmutter, an HPE Cray EX system based on NVIDIA A100 GPUs and AMD Milan CPUs, from Cori, a Cray XC40 system based on Intel Haswell and KNL processors.

VASP - #1 Production Code at NERSC

- VASP, a widely used materials science code, uses over 20% of NERSC computing cycles.
- VASP has been ported to GPUs with OpenACC and is highly optimized for NVIDIA GPUs.
- **Yet, the build and runtime options must be explored to get the best performance out of Perlmutter.**

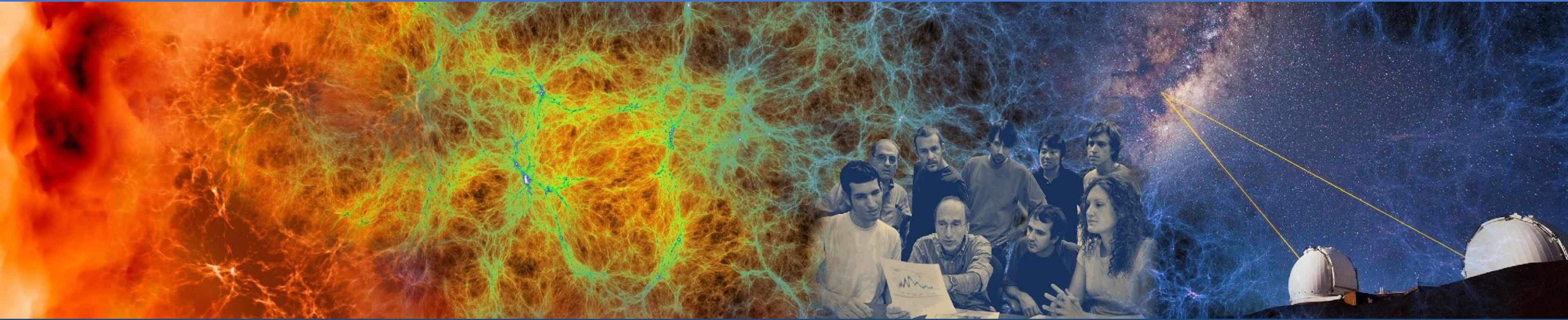


Machine time breakdown by applications (AY 2018)

Our Work and Contributions

- Our work:
    - Explore runtime options (both system and internal to VASP) to get the best performance out of Perlmutter and derive the best practice tips for users
- Contributions:
    - Provide performance guidance for hundred of VASP users on Perlmutter through its lifetime (~ 5 years) - ensuring 20% of Perlmutter computing cycles are used efficiently.
    - Provide feedback to HPC architects and others about how this new system performs for a real-world scientific code with a huge user base.
    - Provide reference data for future system procurements.

System Configuration and Experimental Setup

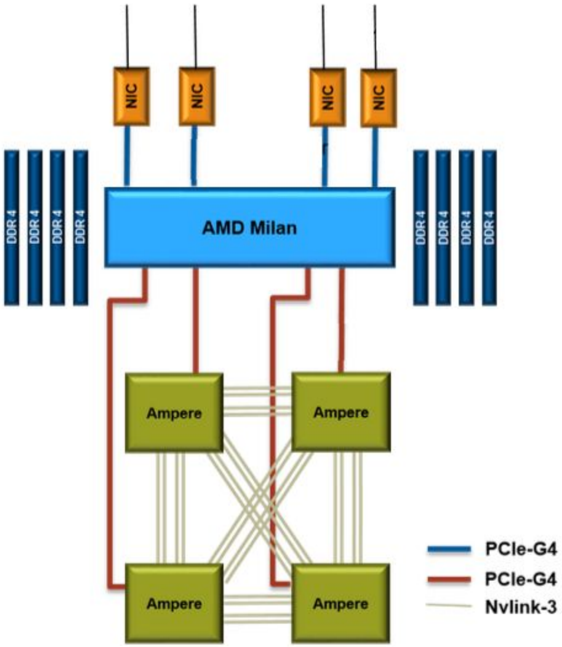


Perlmutter Compute Nodes

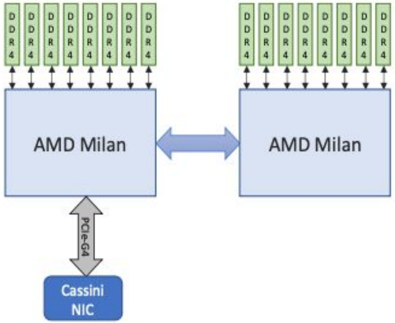
- Single AMD EPYC 7763 (Milan) CPU
- 64 cores per CPU
- Four NVIDIA A100 (Ampere) GPUs
- PCIe 4.0 GPU-CPU connection
- PCIe 4.0 NIC-CPU connection
- 4 HPE Slingshot 11 NICs
- 256 GB of DDR4 DRAM
- 40 GB of HBM per GPU with:
- 1555.2 GB/s GPU memory bandwidth
- 204.8 GB/s CPU memory bandwidth
- 12 third generation NVLink links between each pair of GPUs
- 25 GB/s/direction for each link

Data type	GPU TFLOPS
FP32	19.5
FP64	9.7
TF32 (tensor)	155.9
FP16 (tensor)	311.9
FP64 (tensor)	19.5

GPU Node



CPU Node



- 2x AMD EPYC 7763 (Milan) CPUs
- 64 cores per CPU
- AVX2 instruction set
- 512 GB of DDR4 memory total
- 204.8 GB/s memory bandwidth per CPU
- 1x HPE Slingshot 11 NIC
- PCIe 4.0 NIC-CPU connection
- 39.2 GFlops per core
- 2.51 TFlops per socket
- 4 NUMA domains per socket (NPS=4)

VASP

- Ab Initio materials science code, solving iteratively

  $$[-\frac{1}{2}\nabla^2 + V(\mathbf{r})]\Psi_i(\mathbf{r}) = \epsilon_i\Psi_i(\mathbf{r}), i = 1, 2, ..., N$$

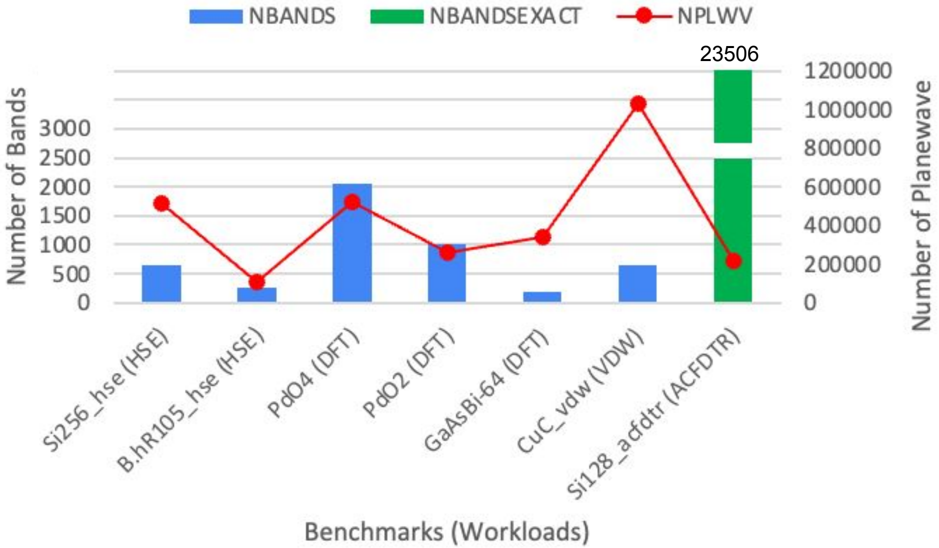
  $\mathbf{\Psi}_{\mathbf{i}}(\mathbf{r})$, ”one-electron” orbitals, expanded in a planewave basis; ε_i, energies; N, the number of orbitals or bands

- VASP code
  - Written mainly in Fortran 90 and heavily utilizes FFTs and Linear Algebra libraries
  - Parallelized with MPI and OpenMP for multi-/many-core CPUs and MPI and OpenACC for GPUs in a single source
  - OpenACC Optimizations include using NCCL for GPU communications, batching FFTs, re-ordering loops, using OpenACC asynchronous execution queues, etc. In addition, the CUDA-aware MPI and optimized math libraries are utilized.

Configurations and Benchmark Approach

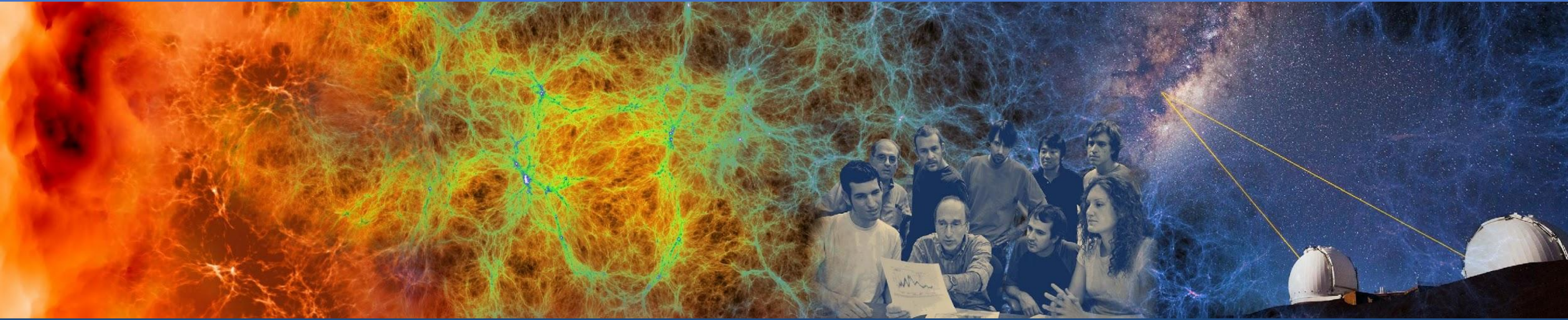
- VASP 6.4.1 (OpenACC and OpenMP ports)
- Compiled with NVIDIA compiler 22.7
- Libraries:
  - Cray MPICH 8.2.5; MKL from Intel oneAPI 23.0.0 and its FFTW3 wrappers to FFT; CUDA 11.7, QD, CUBLAS, CUSOLVER, CUFFT from HPCSDK 22.7; NCCL 2.15.5; HDF5 1.12.2.
- Perlmutter runs HPE Cray OS 2.4 (SLES15SP4) and Slurm 22.05.8
- Used LOOP+ time to measure performance where applicable, and elapsed time for the ACFDTR and energy/power tests
- Repeated each test 5-10 times and selected the best

Benchmarks Were Chosen to Cover the Representative VASP Workloads and to Exercise Different Code Paths



- Based on the VASP user survey at NERSC in 2023
- Designed to reflect day-to-day scientific runs instead of heroic runs

VASP Performance on Perlmutter GPUs and CPUs

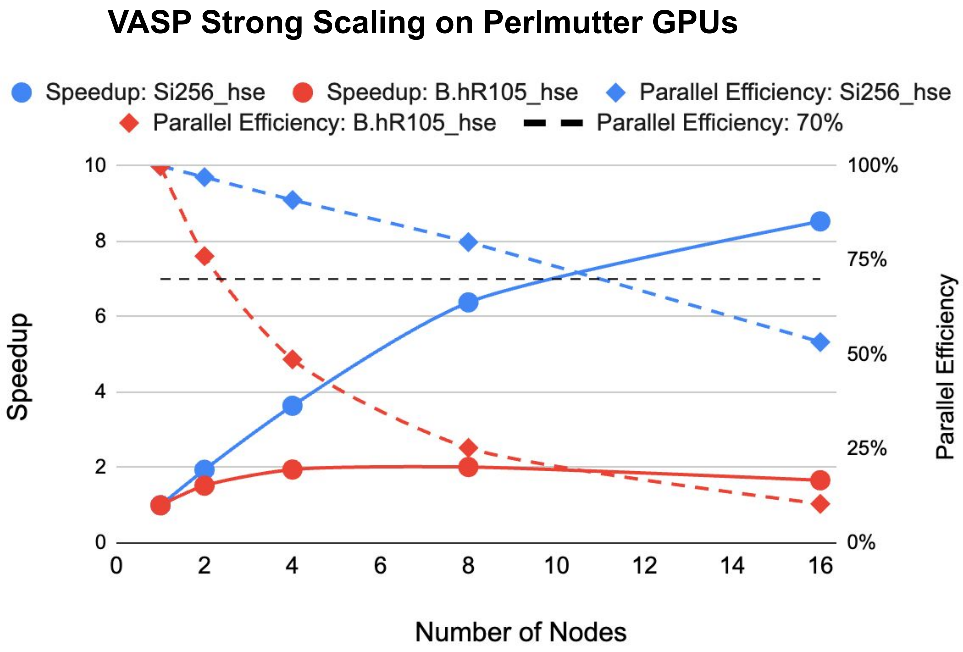


Questions Addressed

- Running on GPUs
  - How many nodes/GPUs are optimal for VASP jobs on Perlmutter GPUs?
  - Do extra OpenMP threads help VASP performance?
  - How much performance gain is there from using NCCL?
  - Does MPS help VASP performance?
  - Does the number of NICs per node affect VASP performance?
- Running on CPUs
  - What is the optimal number of threads per task?
  - Does SMT help VASP performance?
  - How many nodes are optimal for VASP jobs on Perlmutter CPUs?
- GPU speedup over CPUs

https://cug.org/proceedings/protected/cug2023_proceedings/includes/files/pap130s2-file1.pdf

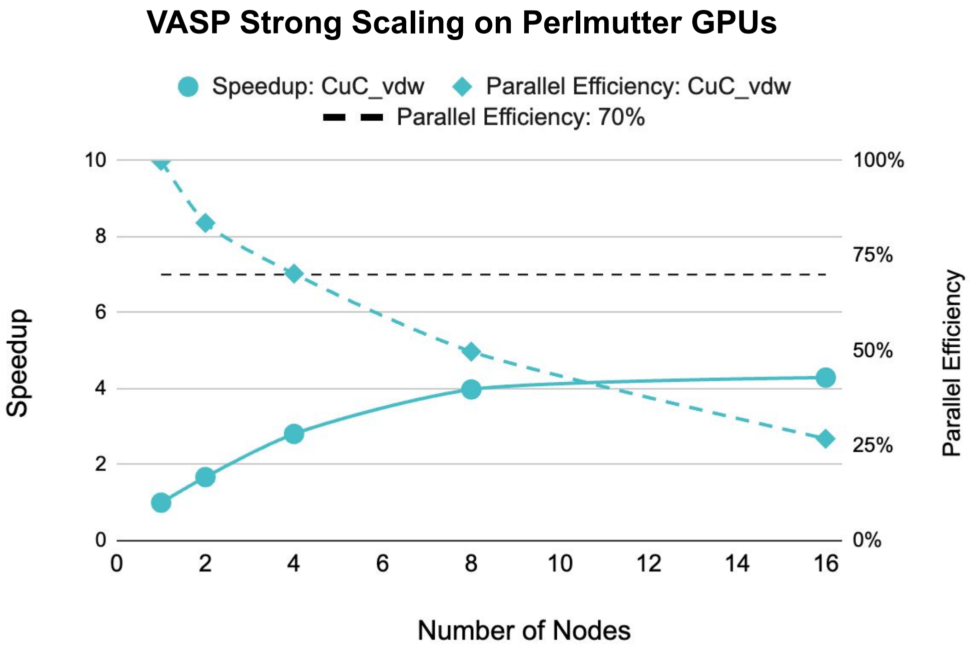
Selecting the Number of GPU Nodes for VASP HSE Workloads



Si256_hse: 640 bands
B.105hR_hse: 256 bands

- VASP parallel scaling depends on the problem sizes.
- For Si256_hse, VASP scales to 8 nodes with more than 75% parallel efficiency.
- 100 bands per node is a reasonable estimate for large HSE workloads.

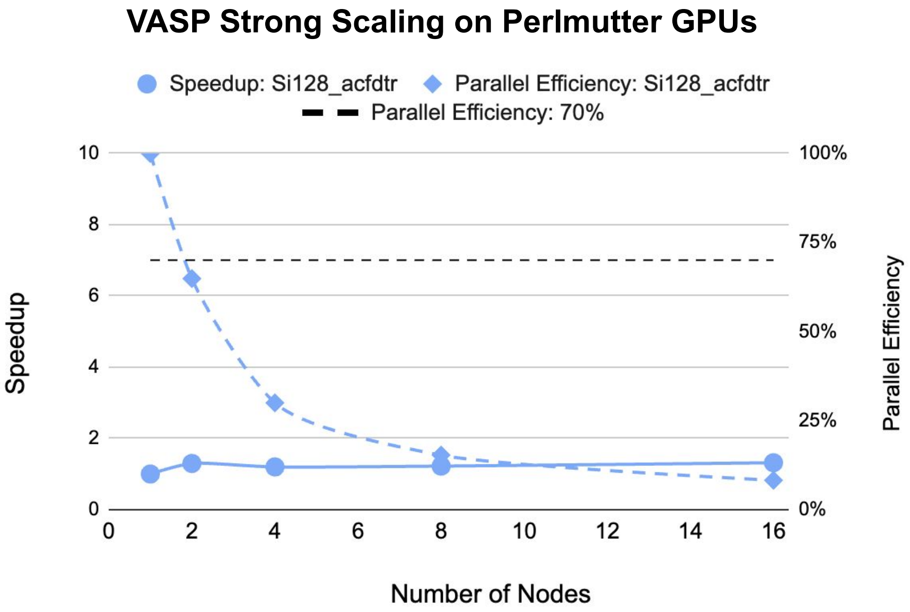
Selecting Number of GPU Nodes for VASP VDW Workloads



CuC_vdw: 640 bands

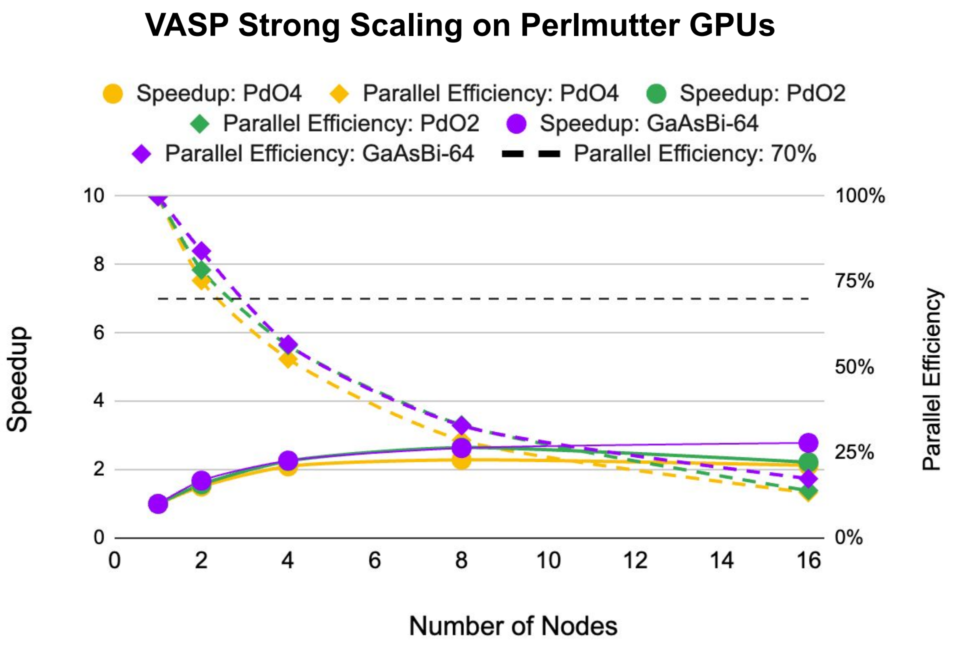
- VASP scales to four nodes with 70% parallel efficiency with CuC_vdw.
- For large VDW workloads, ~200 bands per node is a reasonable estimate.

Selecting the Number of GPU Nodes for VASP ACFDTR Workloads



- VASP does not scale with ACFDTR workloads (memory bottleneck).
- Therefore, one node and 80 GB memory GPU nodes are recommended for larger ACFDTR jobs.
- The memory bottleneck will be removed in future releases of VASP.

Selecting the Number of GPU Nodes for VASP DFT Workloads

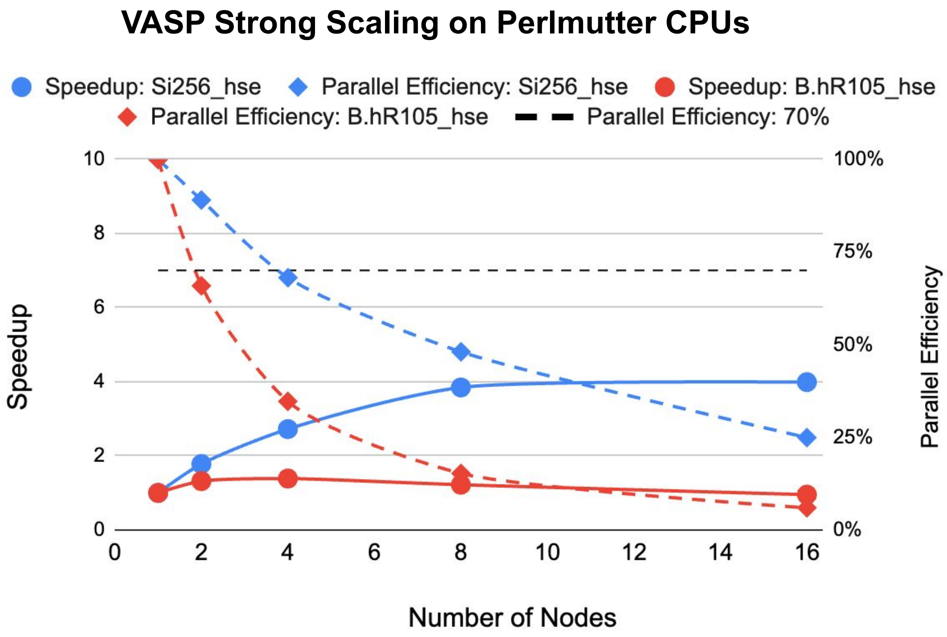


PdO4: 2048 bands
PdO2: 1024 bands

GaAsBi-64:
196 bands and 2 k-point groups

- VASP does not scales over two nodes with PdO4, PdO2, and GaAsBi-64 benchmarks.
- One or two nodes are sufficient for systems containing up to ~300 atoms.

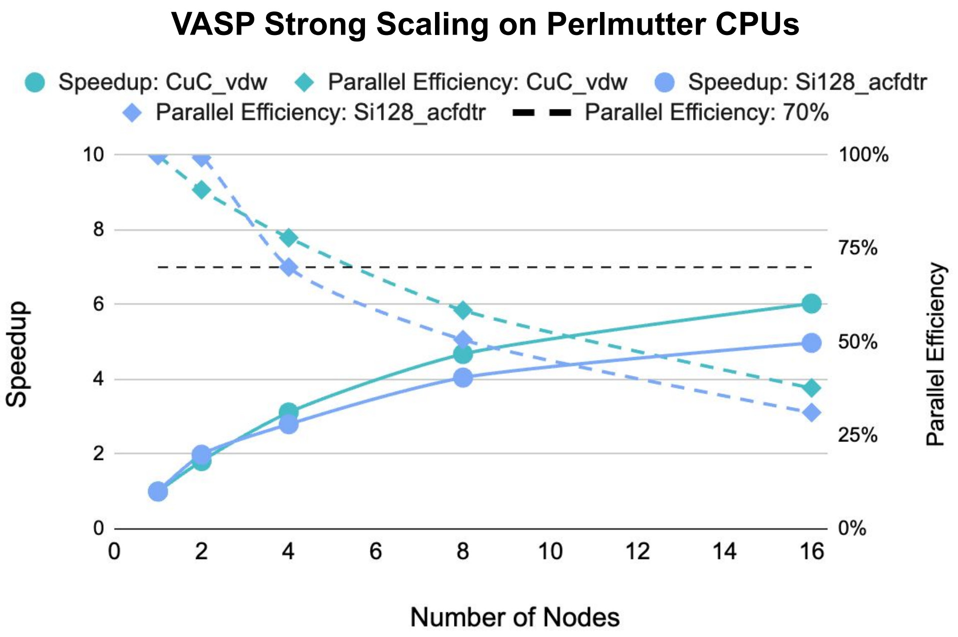
Selecting the Number of Milan CPU Nodes for VASP HSE Workloads



Si256_hse: 640 bands
B.105hR_hse: 256 bands

- VASP parallel scaling depends on the problem sizes.
- For Si256_hse, VASP scales to four nodes with 68% parallel efficiency.
- 200 bands per node is a reasonable estimate for large HSE workloads.

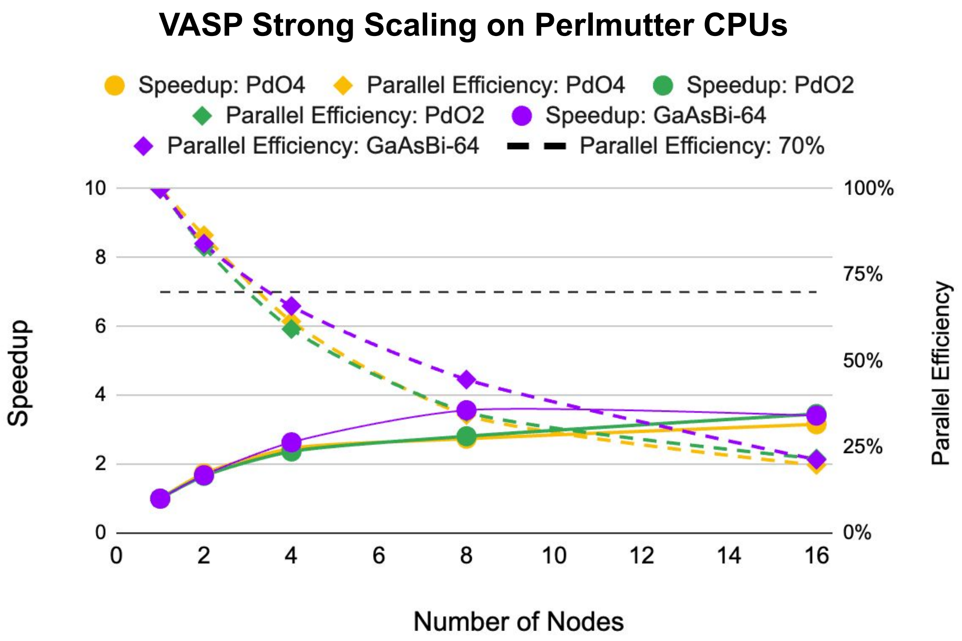
Selecting the Number of Milan CPU Nodes for VASP VDW and ACFDTR Workloads



CuC_vdw:
NBANDS=640
Si128_acfdtr:
NBANDSEXACT=23506

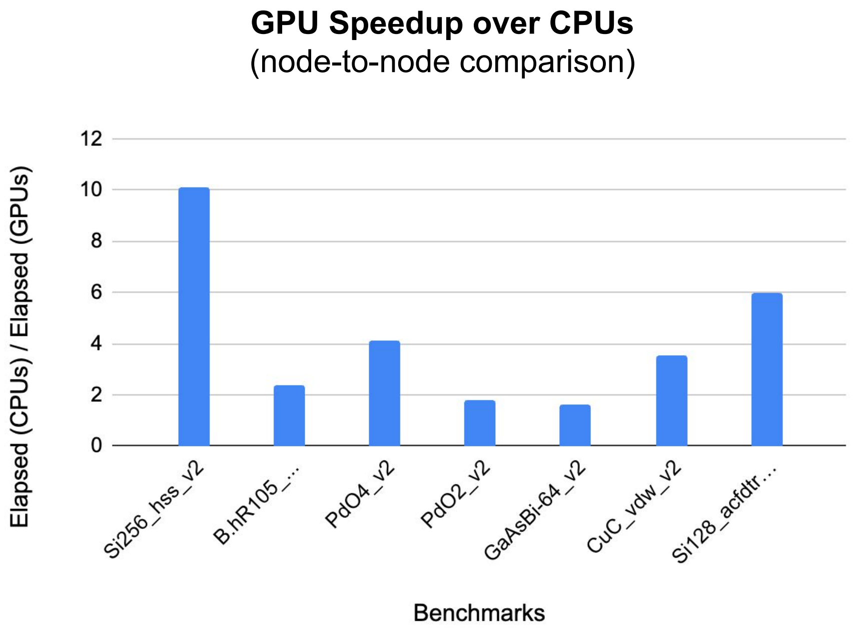
- VASP scales to four nodes for CuC_vdw - roughly 200 bands per node.
- For ACFDTR, VASP scales to ~ four nodes - approximately 6000 bands or planewaves per node.

Selecting the Number of Milan CPU Nodes for VASP DFT Workloads



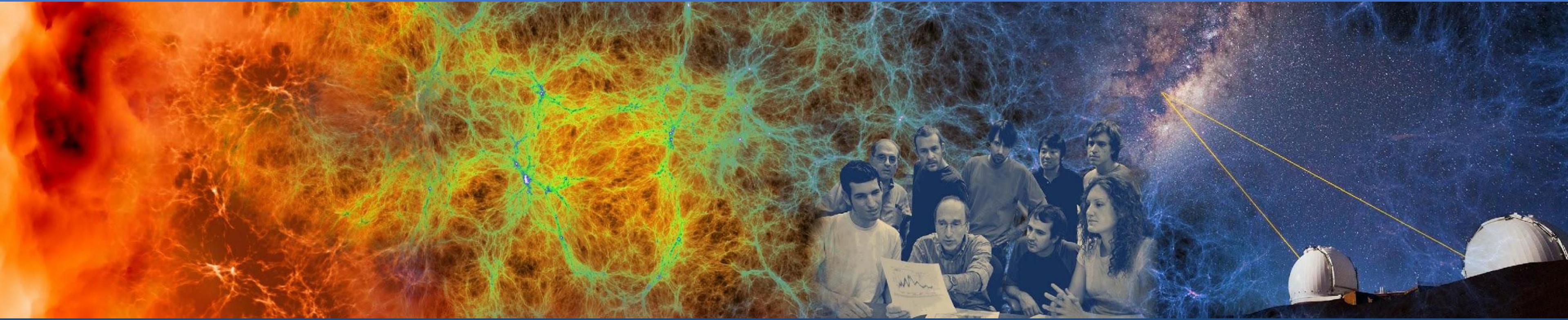
- VASP does not scales over two nodes with PdO4, PdO2, and GaAsBi-64.
- One or two nodes are sufficient for systems containing up to ~300 atoms.

VASP Speedup on Perlmutter GPUs Over CPUs

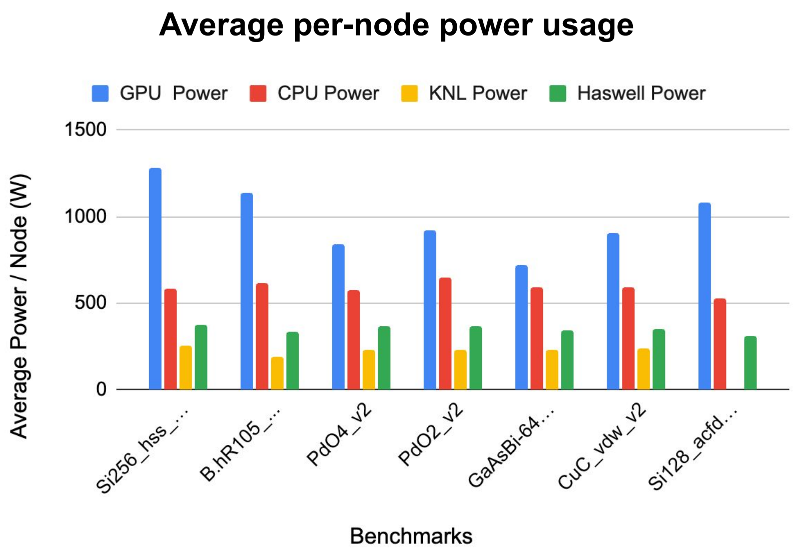


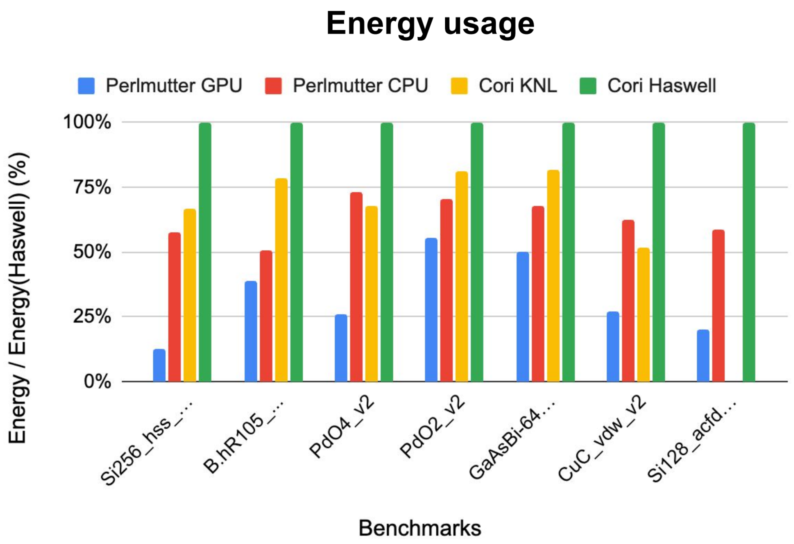
- VASP runs much faster on GPUs compared to running on CPUs; the speedup is 1.8x-10x times depending on workloads and problem sizes
- Larger DFT and HSE, VDW, and ACFDTR workloads get the most benefits running on GPUs
- Running on GPUs incurs significantly less charging.

Energy Efficiency



VASP Power/Energy Usage on Perlmutter and Cori





- VASP uses the most power on GPUs, but the least total energy
- The energy saving from a substantial decrease in time to solution

Summary

1. One or two nodes would be sufficient for systems containing up to several hundred atoms (e.g., 300 atoms) on Perlmutter GPUs and CPUs.
    a. DFT workloads do not scale over more than two nodes for ~300 atom systems
    b. For the HSE and VDW workloads that scale better, roughly distributing 100 - 200 bands per node on GPUs; 200 bands or more on CPUs
    c. ACFDTR workloads do not scale on GPUs, so use one GPU node; for larger problems, use the 80 GB GPU nodes. ACFDTR scales to multiple nodes on CPUs - approximately 6000 bands or planewaves per node.
2. **Running VASP on GPUs gets the results faster, reduces charging, and saves energy.**

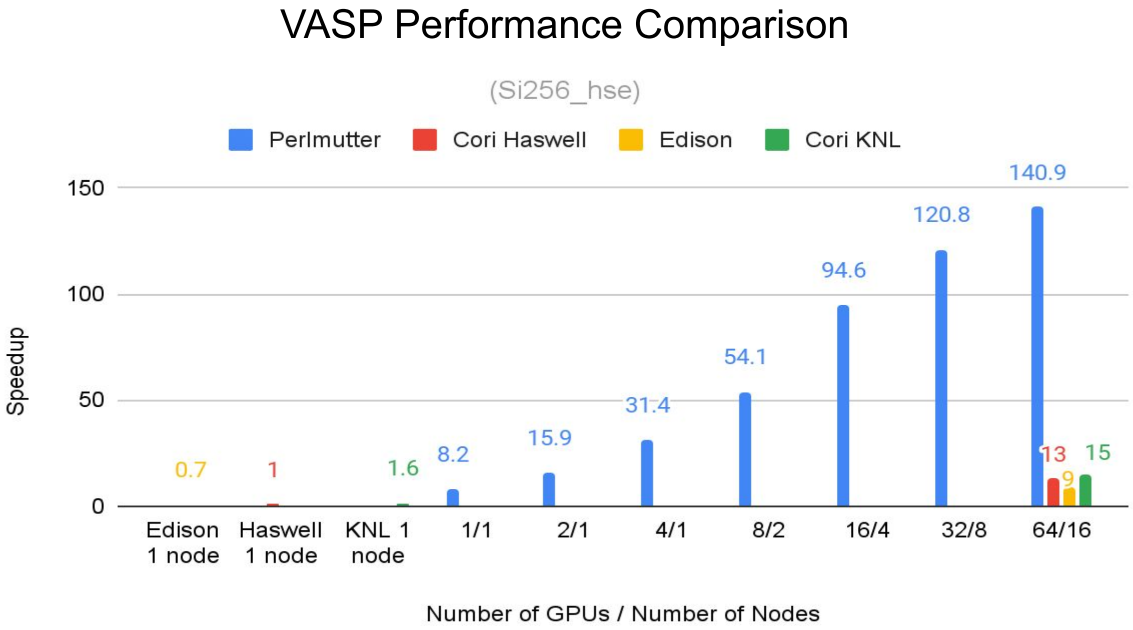
Acknowledgement

- This work was supported by the Office of Advanced Scientific Computing Research in the Department of Energy Office of Science under contract number DE-AC02-05CH11231. It used the resources of the National Energy Scientific Computing Center (NERSC) at the Lawrence Berkeley National Laboratory.
- The VASP users at NERSC participated in the VASP usage survey in 2023, from which the representative VASP workloads were derived.
- Our NERSC colleagues: Nick Wright for valuable input and advice; Sridutt for providing power analysis scripts and help; Taylor for a pointer to Slingshot options in Cray MPICH; all members of the Advanced Technologies Group for valuable feedback.

Thank you!

Backup Slides

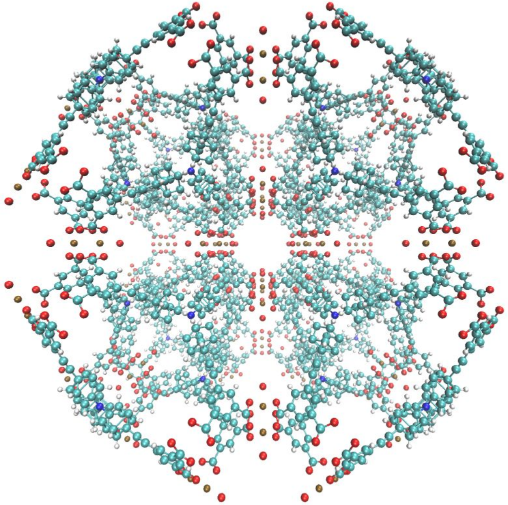
VASP has Been Highly Optimized for NVIDIA GPUs



Perlmutter GPUs speed up VASP substantially!

NESAP FOM

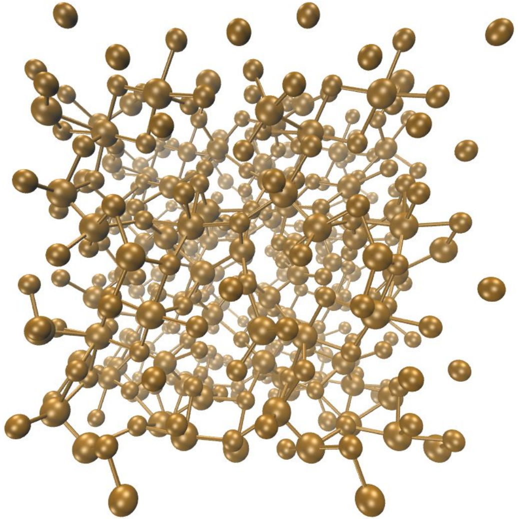
Perlmutter GPUs Enable Heroic VASP Runs



MOF structure (3584 atoms)

Completed 99 ionic steps in 32.8 hours using 32 nodes (128 GPUs) on Perlmutter

Electrons(Ions): 13408 (3584); Functional: DFT; Algo: CG (BD+RMM); NBANDS=   9600; FFT grids: 378x378x378; 756x756x756; NPLWV: 54,010,152; KPOINTS: 1 1 1



Amorphous VF3 (416 atoms)

Completed 66 ionic steps in 17.2 hours using 160 nodes (640 GPUs) on Perlmutter

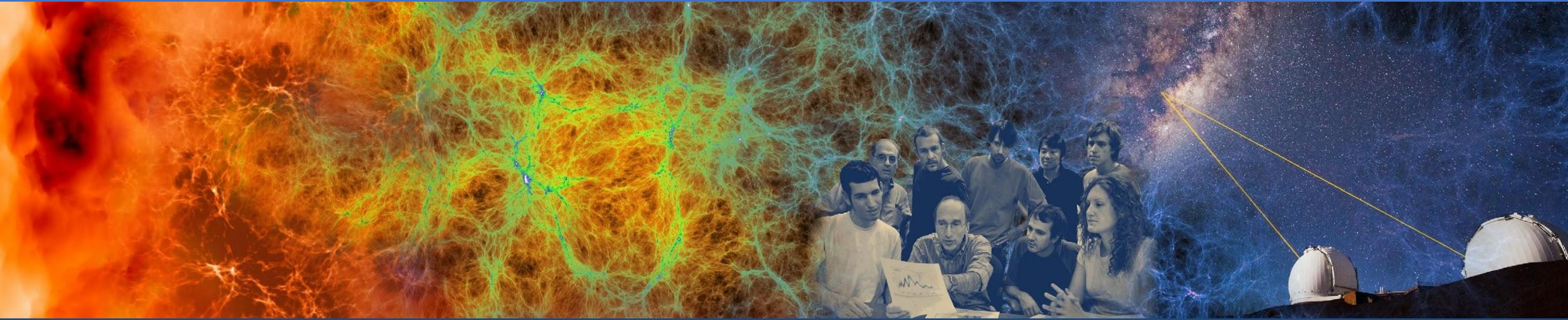
Electrons (Ions): 2592 (416); Functional: DFT; Algo: CG (BD+RMM); NBANDS: 1920; NPLWV: 1,905,120 ; FFT grids: 126x126x120; 192x180x180; KPOINTS: 7 7 7 (172 irreducible kpoints)

Benchmarks Were Chosen to Cover the Representative VASP Workloads and to Exercise Different Code Paths

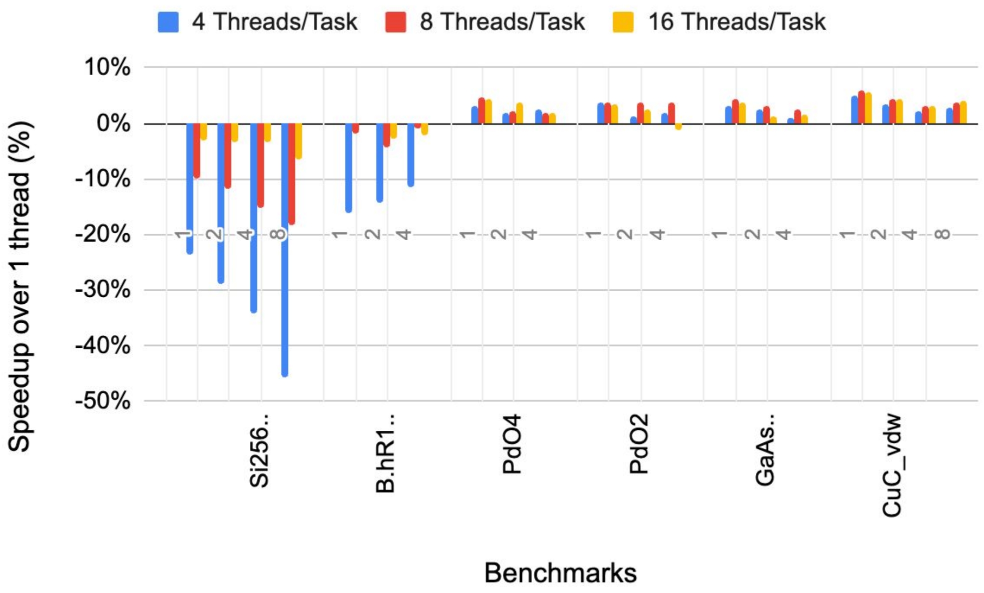
	Si256_hse	B.hR105_hse	PdO4	PdO2	GaAsiBi-64	CuC_vdw	Si128_acfdtr
Electrons (Ions)	1020 (255)	315 (105)	3288 (348)	1644 (174)	266 (64)	1064 (98)	512 (128)
Functional	HSE	HSE	DFT	DFT	DFT	VDW	ACFDTR/
Algo	CG (Damped)	CG (Damped)	RMM (VeryFast)	RMM (VeryFast)	BD+RMM (Fast)	RMM (VeryFast)	ACFDTR
NBANDS (NBANDSEXACT)	640	256	2048	1024	192	640	324 (23506)
FFT grids	80x80x80 160x160x160	48x48x48 96x96x96	80x120x54 160x240x108	80x60x54 160x120x108	70x70x70 140x140x140	70x70x210 120x120x350	60x60x60 120x120x120
NPLWV	512000	110592	518400	259200	343000	1029000	216000
KPOINTS (KPAR)	111	111	111	111	444 (2)	331	111

- Based on the VASP user survey at NERSC in 2023
- Designed to reflect day-to-day scientific runs instead of heroic runs

VASP Performance on GPU Nodes

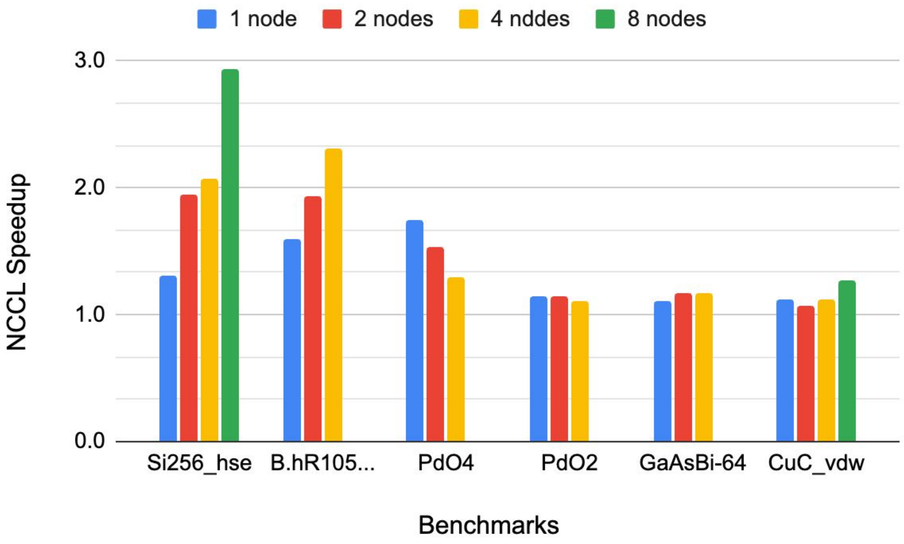


Performance Effect of OpenMP Threads for VASP Jobs Running on GPUs



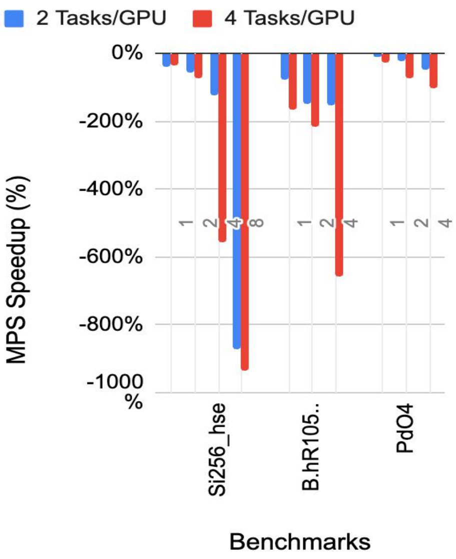
- Benefit DFT workloads slightly
- Slow down HSE workloads significantly

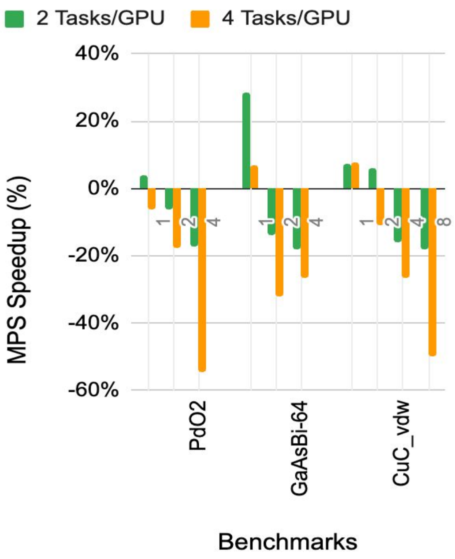
VASP Performance Effect of NCCL



- NCCL boosts VASP performance significantly; the most benefit is observed with the HSE workloads
- NCCL should be used whenever possible

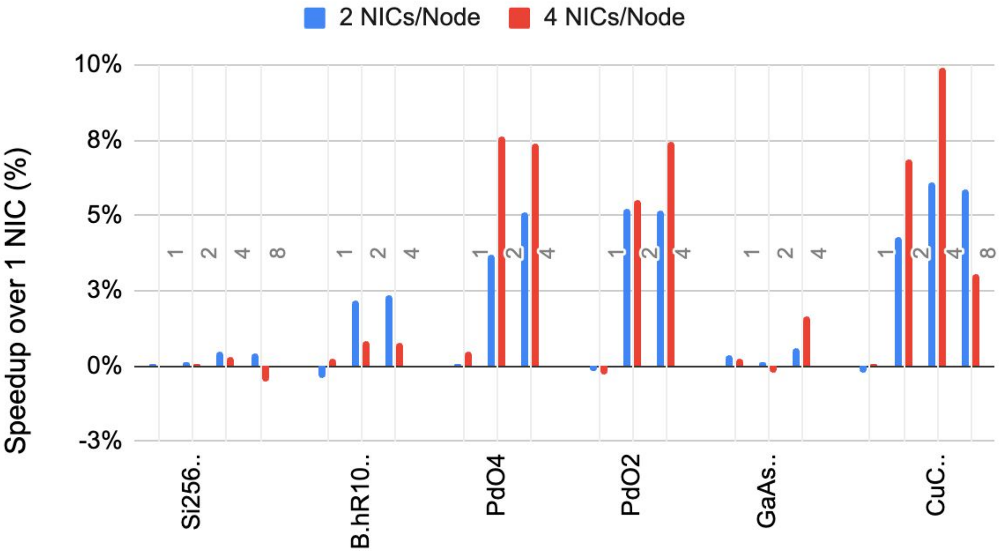
MPS and VASP Performance





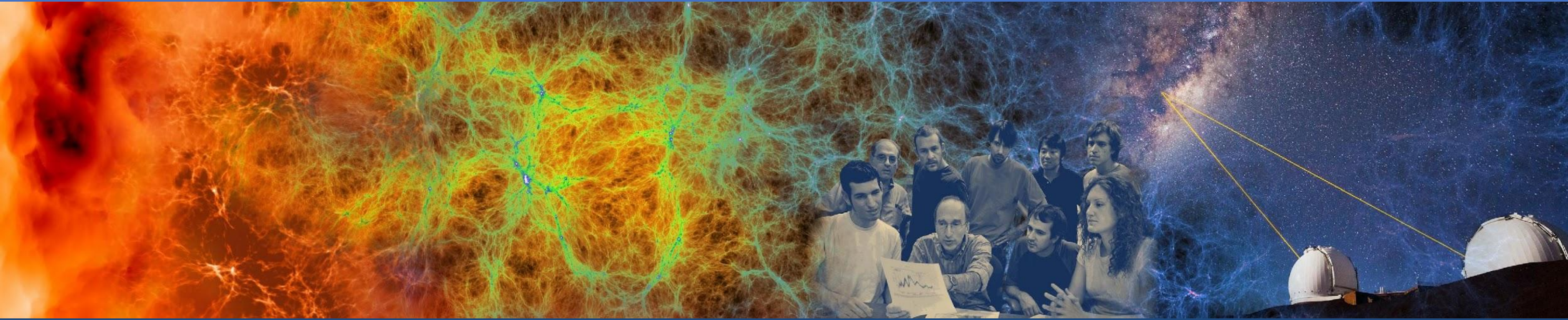
- MPS slows down VASP significantly
- May enable small systems with many k-points on GPUs (LUSENCCL=.FALSE)

Number of NICs per Node

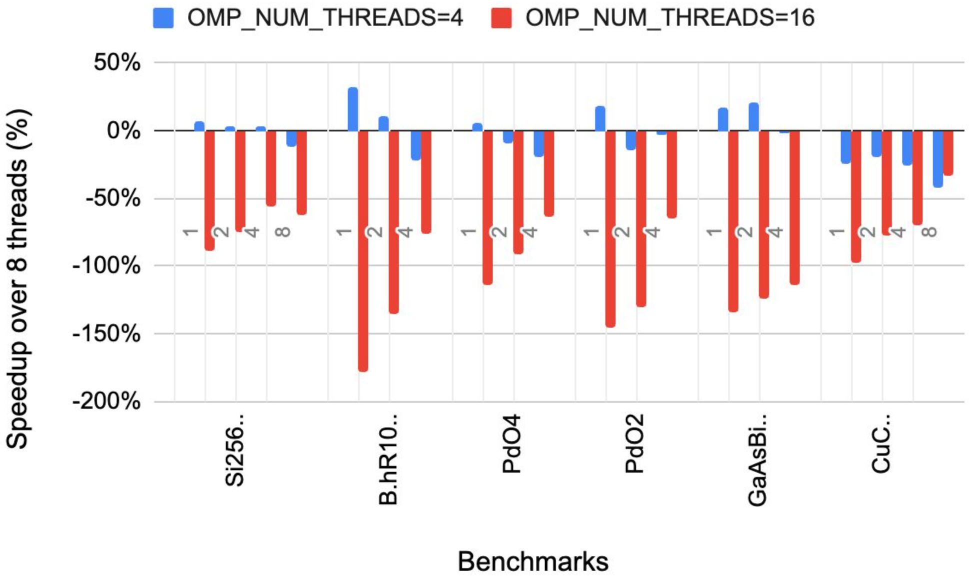


- The number of NICs per node has a minor performance impact on VASP.
- Using default four NICs per node is recommended.

VASP Performance on Perlmutter CPU Nodes

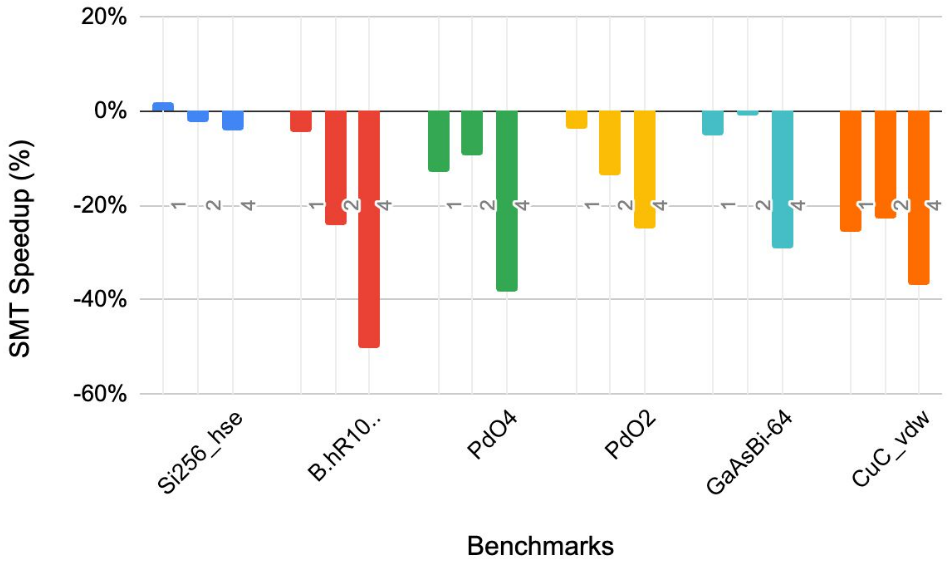


Thread Performance on Milan CPUs



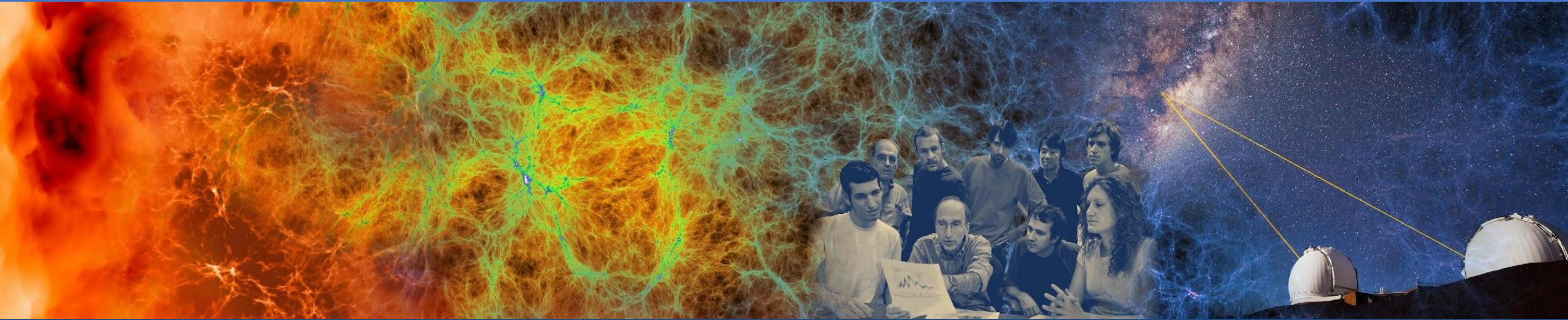
- Eight threads per task is a safe choice in practice
- Using four or fewer threads may help performance at small node counts.

Simultaneous Multi-Threading (SMT)

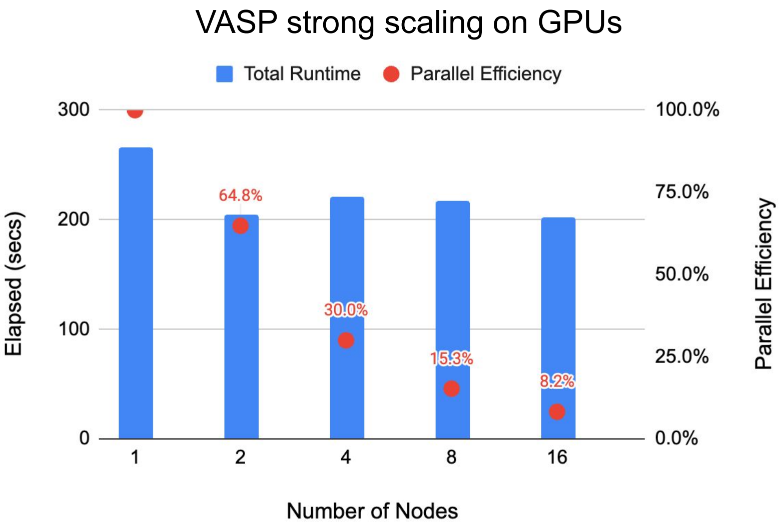


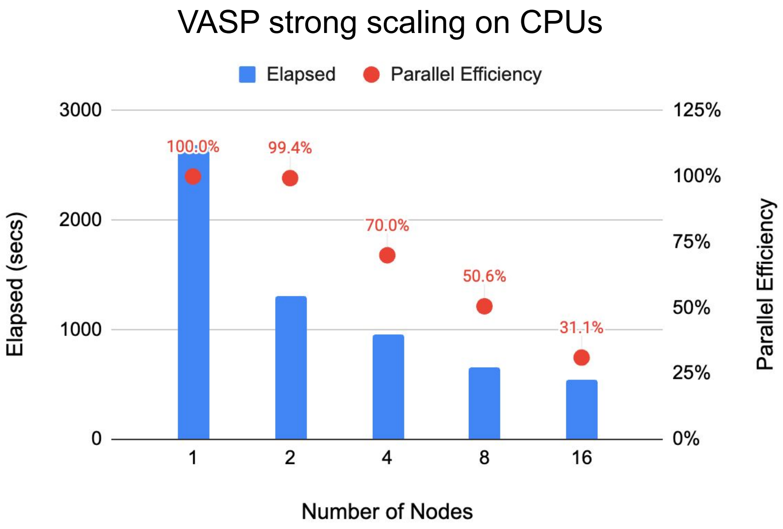
SMT does not help VASP performance but benefits HSE workloads slightly at node count one.

ACFDTR Workloads



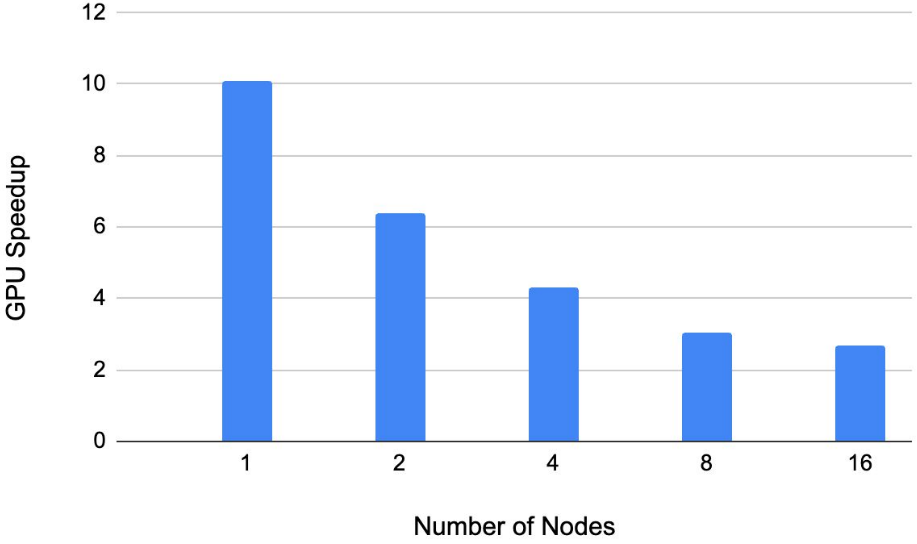
VASP Performance With ACFDTR Workloads





VASP ACFDTR GPU port does not scale well over multiple GPU nodes yet, while the hybrid OpenMP+MPI VASP for CPUs scales better over multiple CPU nodes

VASP Speedup on GPUs over CPUs

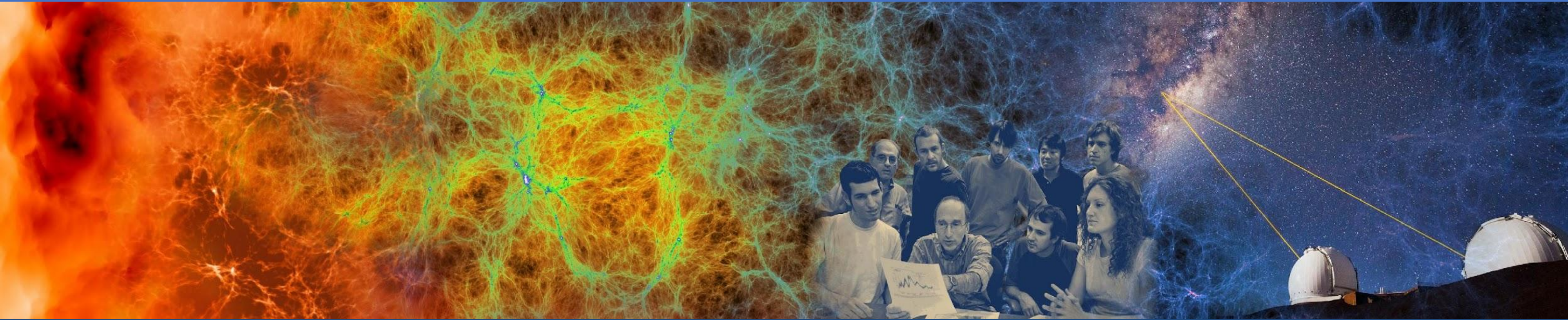


Running on GPUs reduces the time to solution for the ACFDT workloads significantly.

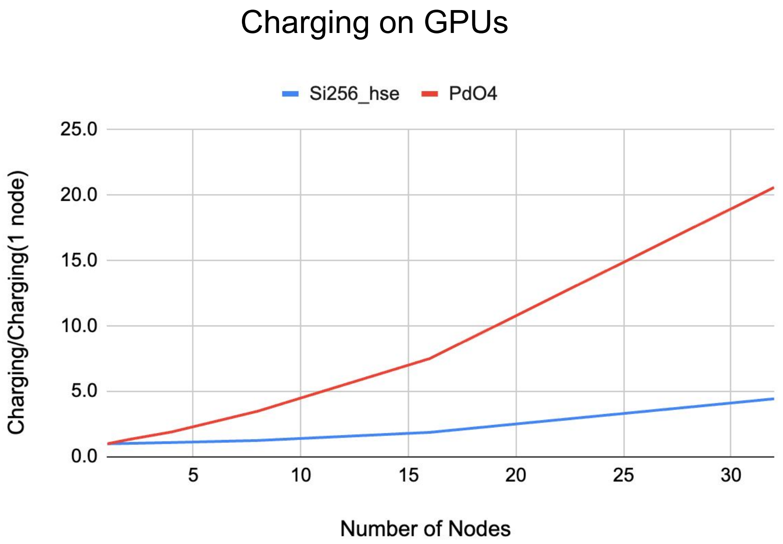
ACFDTR Workloads

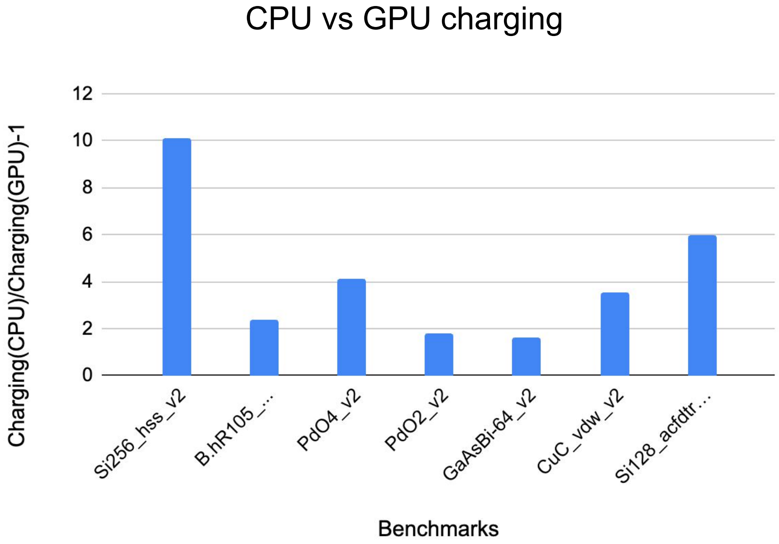
- The compute-intensive ACFDTR algorithm has been ported to GPUs, significantly reducing the time to solution.
- While the hybrid OpenMP+MPI code scales with multiple nodes on CPUs, the GPU port of ACFDTR hardly scales to multiple nodes. However, the GPU port is significantly faster than the CPU port; therefore, running ACFDTR workloads on GPUs is recommended.
- The ACFDTR GPU implementation is memory intensive. Consider using the 80GB GPU nodes for systems requiring more significant memory.
    - The memory bottleneck will be removed to enable larger system computation soon.

Charging Efficiency



Charging: Perlmutter CPUs vs GPUs





- Running VASP outside its parallel scaling region results in high charging with little benefit in runtime.
- Running on GPUs incurs significantly less charging.