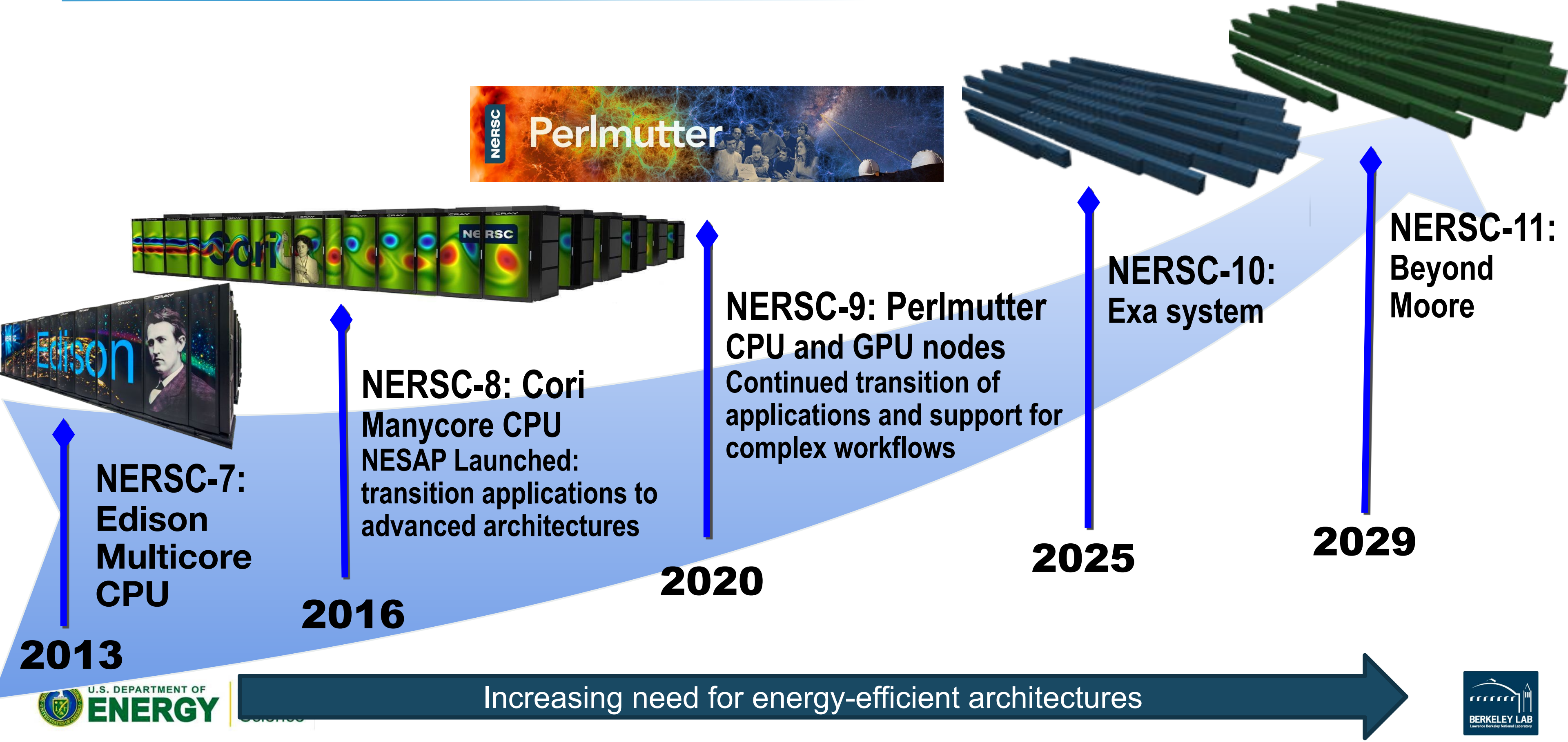
GPUs 101

Jack Deslippe

Application Performance Lead
NERSC

NERSC Systems Roadmap



2013 – NERSC-7: Edison Multicore CPU

2016 – NERSC-8: Cori Manycore CPU
NESAP Launched: transition applications to advanced architectures

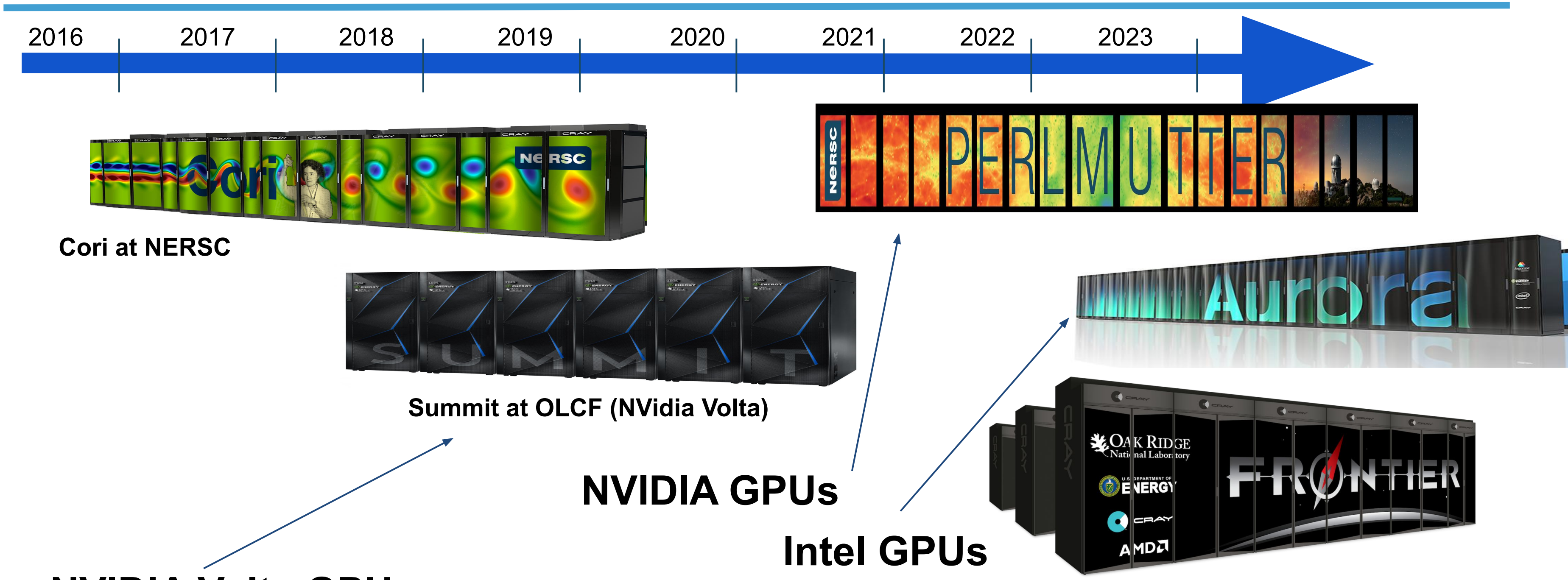
2020 – NERSC-9: Perlmutter CPU and GPU nodes
Continued transition of applications and support for complex workflows

2025 – NERSC-10: Exa system

2029 – NERSC-11: Beyond Moore

Increasing need for energy-efficient architectures

DOE HPC Roadmap



Cori at NERSC

Summit at OLCF (NVidia Volta)

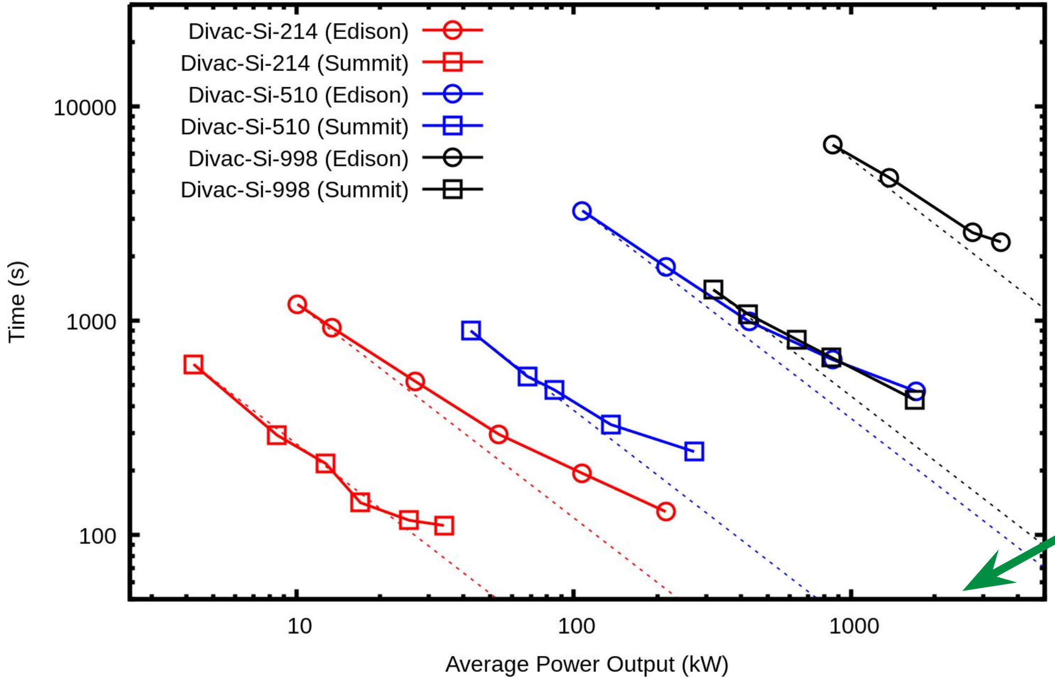
NVIDIA GPUs

Intel GPUs

NVIDIA Volta GPUs

AMD GPUs

Energy Efficiency Across Architectures



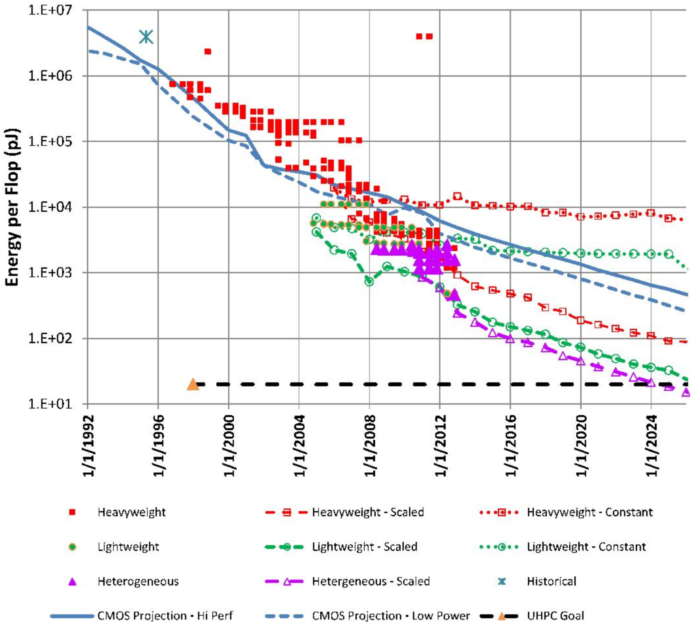
Improving Energy Efficiency

Circles: EDISON@NERSC CPU only

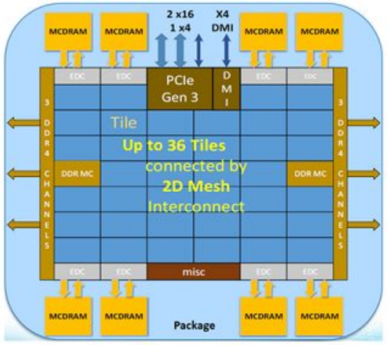
Squares: SUMMIT@OLCF CPU+GPU

Change Has Arrived (Whether you want it to or not)

Driven by power consumption and heat dissipation toward lightweight cores



Knights Landing Overview



KNL: 215-230 W

2-socket Haswell: 270 W

Cori is a boon to science in the U.S. because of new capabilities, but the Intel Xeon Phi many-core architecture requires a code modernization effort to use efficiently.

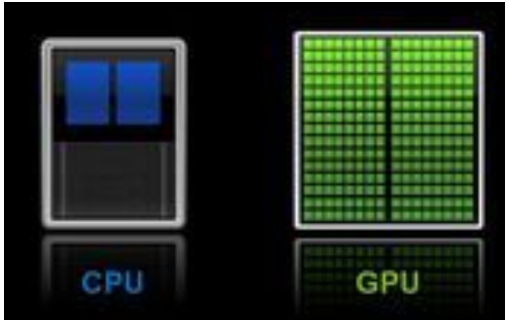
The Main Things You Need to Know About Programming A100 GPUs

1. You Need Lots of Parallelism

CPUs to GPUs

CPU (Haswell)

- 64 cores
- 2 threads each
- 2x256-bit vectors
- double precision
    - ~2000 way parallelism (64*4*8)



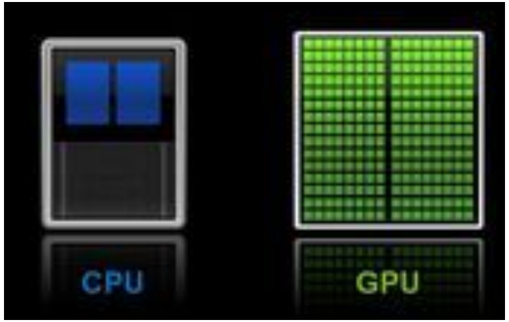
GPU (A100)

- 108 SM
- Up to 64 warps per SM
  (2 active at a time)
- 32 SIMT per warp
- double precision
    - 200,000+ way parallelism (108*64*32)

CPU - Speed

GPU - Throughput

CPUs to GPUs



CPU (Haswell)

- 64 cores
- 2 threads each
- 2x256-bit vectors
- double precision
  - ~2000 way parallelism (64*4*8)

GPU (A100)

- 108 SM
- Up to 64 warps per SM (2 active at a time)
- 32 SIMT per warp
- double precision
  - 200,000+ way parallelism (108*64*32)

CPU - Speed

GPU - Throughput

Oversubscribing GPUs (w/ Warps and Streams) helps hide latency, too!

The Main Things You Need to Know About Programming A100 GPUs

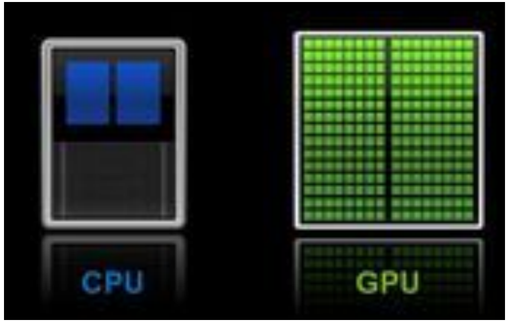
1. You Need Lots of Parallelism

2. A100 GPU Memory is Very Fast. But, moving data to the GPU is Not.

CPUs to GPUs

CPU (Haswell)

- 128GB DDR
- ~120 GB/Sec Memory Bandwidth



GPU (A100)

- 40GB HBM
- 1,500 GB/Sec Memory Bandwidth

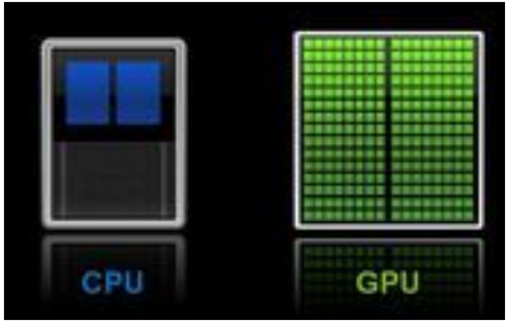
CPU - Speed

GPU - Throughput

CPUs to GPUs

CPU (Haswell)

- 128GB DDR
- ~120 GB/Sec Memory Bandwidth



GPU (A100)

- 40GB HBM
- 1,500 GB/Sec Memory Bandwidth

PCIe ~ 32 GB/Sec

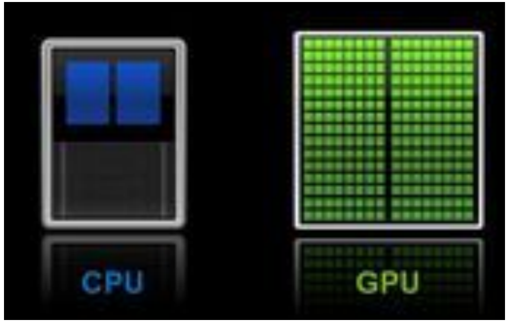
CPU - Speed

GPU - Throughput

CPUs to GPUs

CPU (Haswell)

- 128GB DDR
- ~120 GB/Sec Memory Bandwidth



GPU (A100)

- 40GB HBM
- 1,500 GB/Sec Memory Bandwidth

PCIe ~ 32 GB/Sec

CPU - Speed

GPU - Throughput

Try to avoid moving data back and forth frequently

The Main Things You Need to Know About Programming A100 GPUs

1. You Need Lots of Parallelism

2. A100 GPU Memory is Very Fast. But, moving data to the GPU is Not.

Other Second Order Considerations:

3. There is some overhead in launching kernels. Fusing short kernels together and defining “CUDA Graphs” can help.

The Main Things You Need to Know About Programming A100 GPUs

1. You Need Lots of Parallelism

2. A100 GPU Memory is Very Fast. But, moving data to the GPU is Not.

Other Second Order Considerations:

3. There is some overhead in launching kernels. Fusing short kernels together and defining "CUDA Graphs" can help.

4. HBM is fast, but keeping data in registers, cache and "shared" memory is better!

The Main Things You Need to Know About Programming A100 GPUs

1. You Need Lots of Parallelism

2. A100 GPU Memory is Very Fast. But, moving data to the GPU is Not.

Other Second Order Considerations:

3. There is some overhead in launching kernels. Fusing short kernels together and defining “CUDA Graphs” can help.

4. HBM is fast, but keeping data in registers, cache and “shared” memory is better!
   Find optimal balance between maximizing parallelism and minimizing register spills.

Perlmutter Supports Every GPU Programming Model

	Fortran/ C/C++	CUDA	OpenACC 2.x	OpenMP 5.x	CUDA Fortran	Kokkos / Raja	MPI	HIP	DPC++ / SYCL
NVIDIA	🟩	🟩	🟩	🟩	🟩	🟩	🟩		
CCE	🟩					🟩	🟩		
GNU	🟩	🟩	🟩	🟩		🟩	🟩		
LLVM	🟧	🟧		🟧		🟧	🟧	🟧	🟧

🟩 Vendor Supported

🟧 NERSC Supported

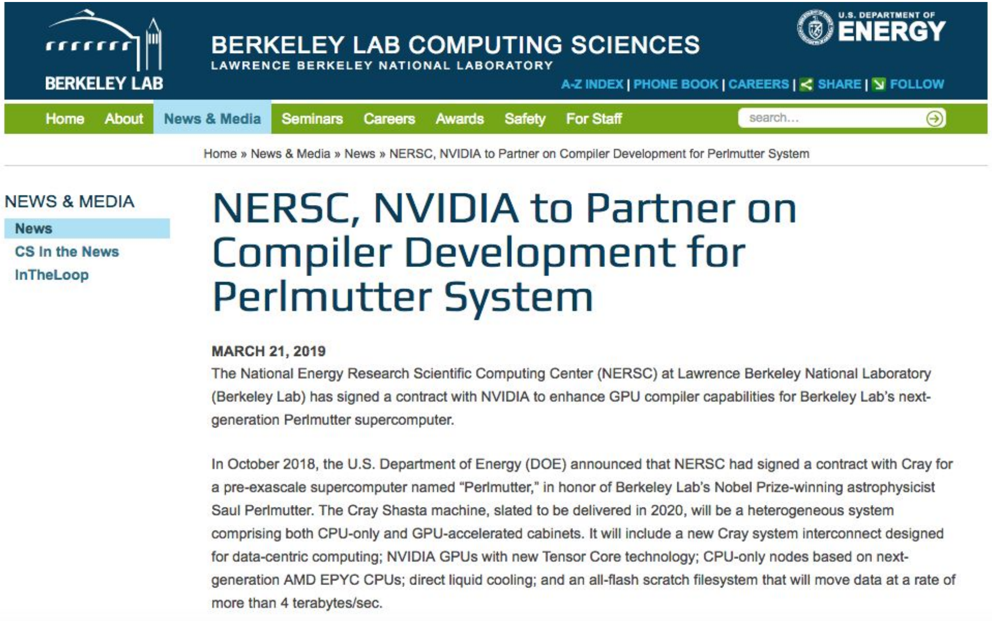
OpenMP NRE partnership with NVIDIA

- Agreed upon subset of OpenMP features to be included in the NVIDIA (was PGI) compiler

- OpenMP test suite created with micro-benchmarks, mini-apps, and the ECP SOLLVE V&V suite

- 5 NESAP application teams partnered with NVIDIA to add OpenMP target offload directives

- The production OpenMP offload compiler was released in April 2021.


BERKELEY LAB
BERKELEY LAB COMPUTING SCIENCES
LAWRENCE BERKELEY NATIONAL LABORATORY
U.S. DEPARTMENT OF ENERGY
A-Z INDEX | PHONE BOOK | CAREERS | SHARE | FOLLOW
Home About News & Media Seminars Careers Awards Safety For Staff
search...
Home » News & Media » News » NERSC, NVIDIA to Partner on Compiler Development for Perlmutter System
NEWS & MEDIA
News
CS In the News
InTheLoop

NERSC, NVIDIA to Partner on Compiler Development for Perlmutter System

MARCH 21, 2019

The National Energy Research Scientific Computing Center (NERSC) at Lawrence Berkeley National Laboratory (Berkeley Lab) has signed a contract with NVIDIA to enhance GPU compiler capabilities for Berkeley Lab's next-generation Perlmutter supercomputer.

In October 2018, the U.S. Department of Energy (DOE) announced that NERSC had signed a contract with Cray for a pre-exascale supercomputer named "Perlmutter," in honor of Berkeley Lab's Nobel Prize-winning astrophysicist Saul Perlmutter. The Cray Shasta machine, slated to be delivered in 2020, will be a heterogeneous system comprising both CPU-only and GPU-accelerated cabinets. It will include a new Cray system interconnect designed for data-centric computing; NVIDIA GPUs with new Tensor Core technology; CPU-only nodes based on next-generation AMD EPYC CPUs; direct liquid cooling; and an all-flash scratch filesystem that will move data at a rate of more than 4 terabytes/sec.

NVIDIA

PGI® COMPILERS & TOOLS

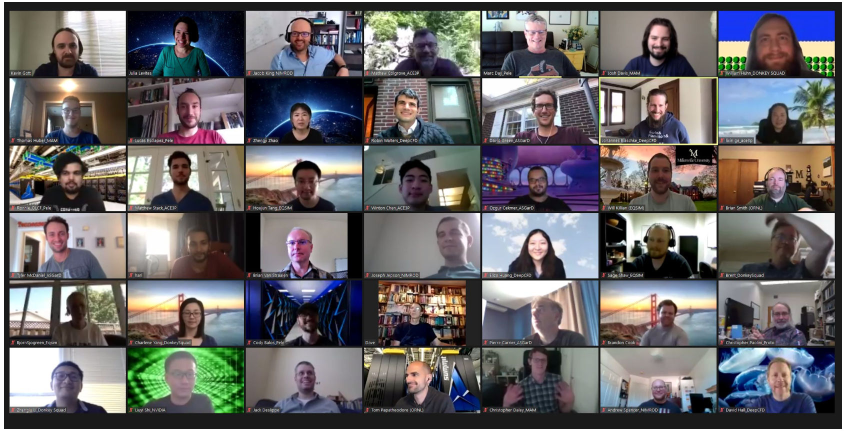
Hackathons

a Hewlett Packard Enterprise company

“Hackathons” have proven to be a highly effective tool for preparing applications for new architectures.

(https://www.gpuhackathons.org) **NERSC provided more team mentors than any other institution to worldwide events in 2020**.

Allows us to reach NERSC teams all around the country and world



NERSC adapted the hackathon format for the COVID work-from-home environment.

Features of this format were popular and effective and we plan to incorporate them into future hackathons.

Broad impact and enablement

Programming models and languages

kokkos

C++

SYCL

ROCm

OpenMP

OpenACC
Directives for Accelerators

Fortran

Community Codes

VASP

LAMMPS

CP2K

WEST!

QUANTUMESPRESSO

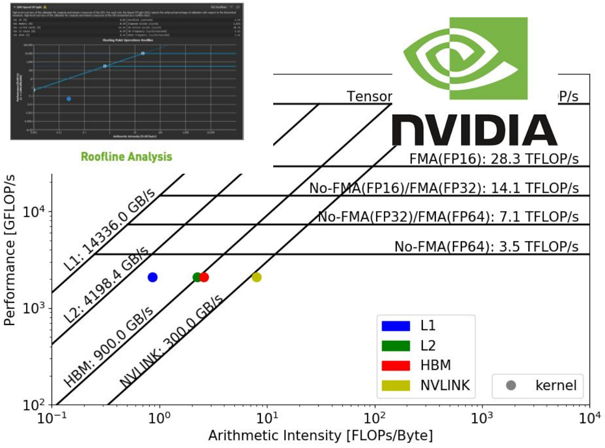
Community Resources

NERSC Documentation

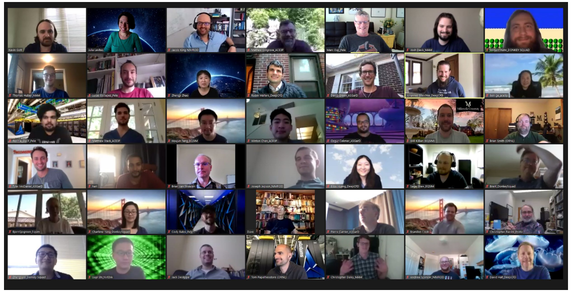
NERSC TRAINING EVENTS

SC20

Vendor tools



Community GPU hack-a-thons



Optimization Challenges For Scientists

Teams often want simple way to wrap their heads around performance when main focus is scientific productivity:

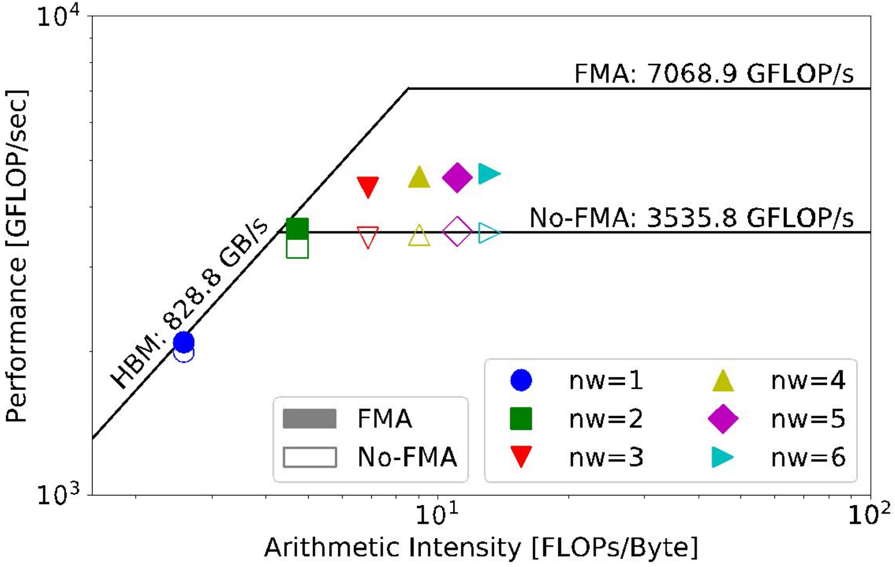
1. Need a sense of absolute performance when optimizing applications.
   - How do I know if my performance is good?
   - Why am I not getting peak performance advertised
   - How do I know when to stop?

2. Many potential optimization directions:
   - How do I know which to apply?
   - What is the limiting factor in my app’s performance?
   - Again, how do I know when to stop?

Roofline on GPUs

nvprof / Nsight can collect all required metrics including data motion from multiple levels of memory hierarchy: L1/Shared, L2, DRAM, *etc*.

Can Plot Roofline Performance Curves within NSight!



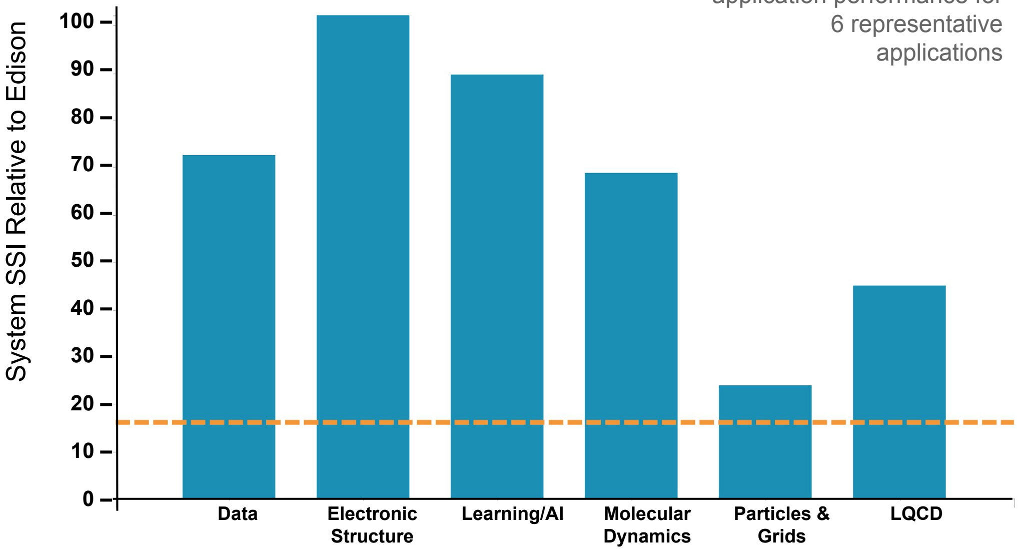
GPP on V100

Projected Application Performance

- We use Perlmutter and Previous GPU performance measurements to estimate/extrapolate a system wide throughput speedup on Perlmutter vs. Edison (the NERSC-7 system).

- Applications from different science areas and algorithmic spaces are able to utilize Perlmutter GPUs

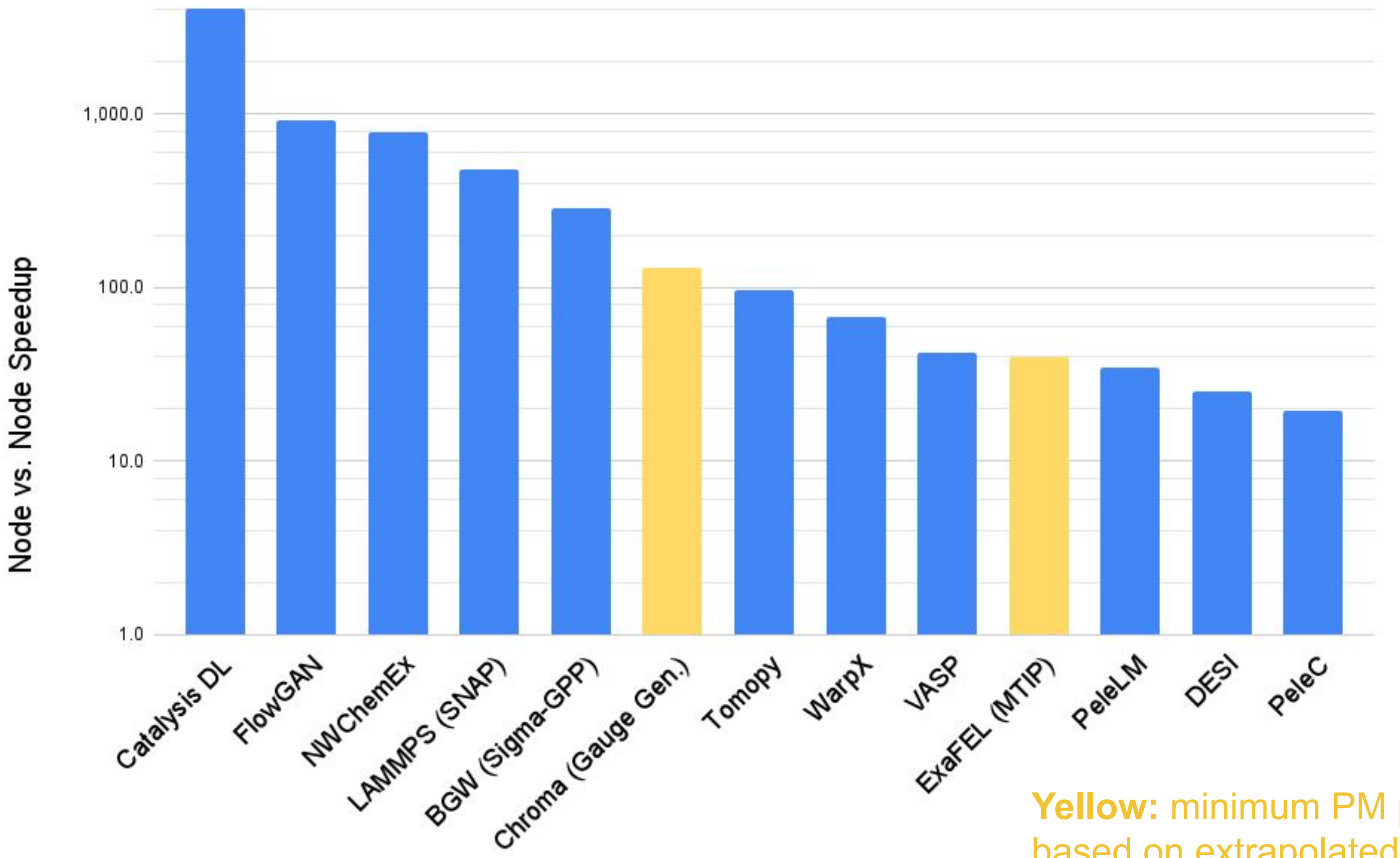
Based on measured application performance for 6 representative applications



Perlmutter System-Wide Performance Performance

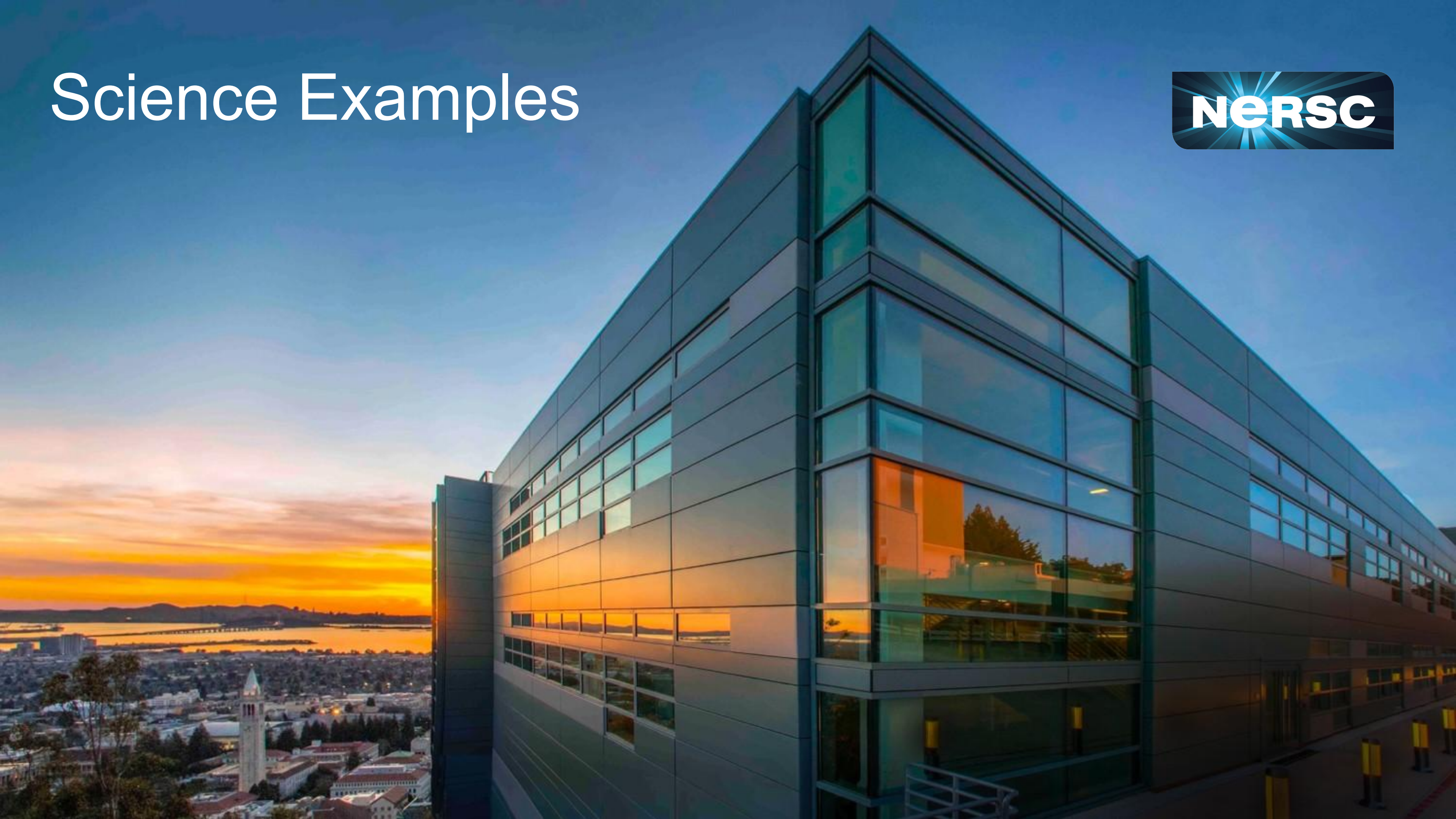
6 applications from different areas of the workload achieve 20X Systemwide throughput increase over Edison.

Perlmutter vs. Edison Node vs. Node Speedups



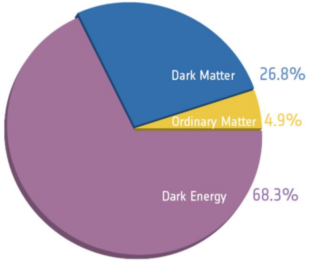
Yellow: minimum PM performance based on extrapolated perf.

Science Examples

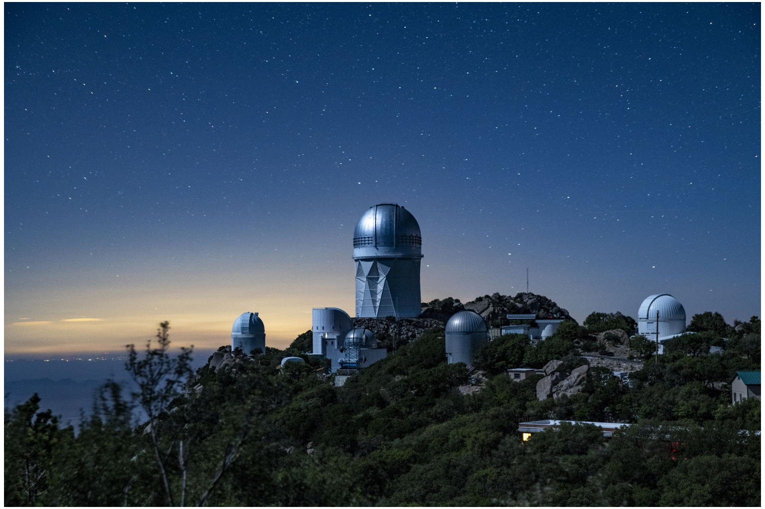


DESI **D**ark **E**nergy **S**pectroscopic **I**nstrument

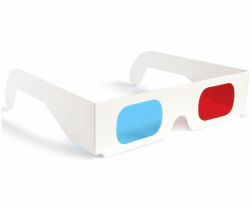
Science: Understand Dark Energy



Scientists believe about 70 percent of the universe is dark energy, although we don't have a good understanding of what it is



The DESI instrument will send NERSC data every night for 5 years

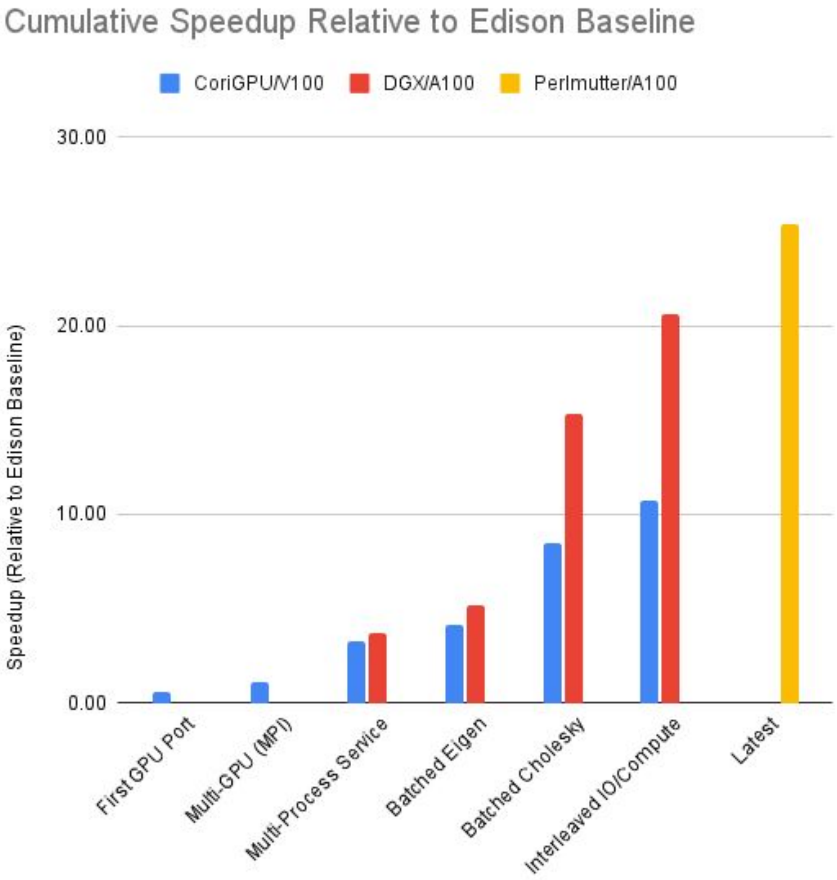


Data will be used to construct the most detailed 3D map of the universe to date and better understand the nature of dark energy

DESI

Dark **E**nergy **S**pectroscopic **I**nstrument

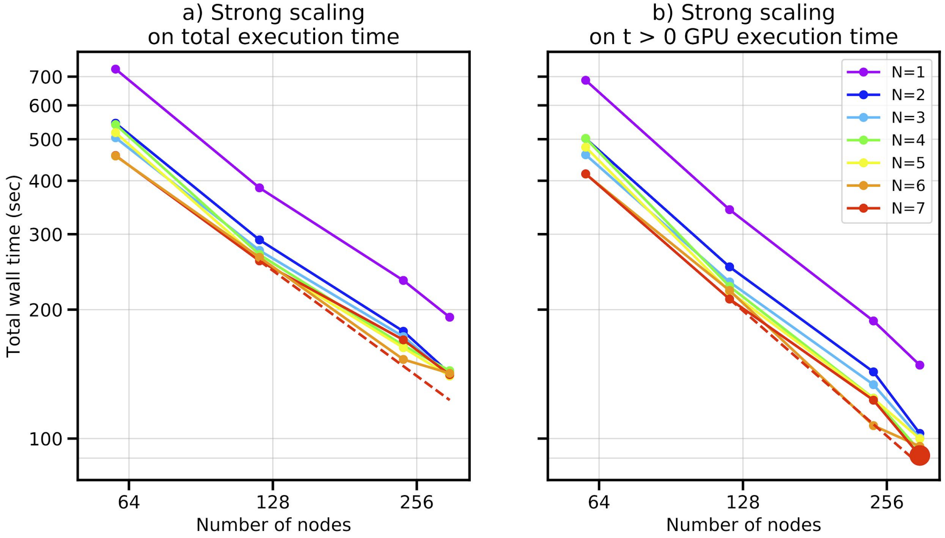
- DESI Spectral Extraction is an image processing code implemented in Python.
- Completed major refactor of optimized CPU code and initial GPU port in early 2020.
- Major optimization milestones include: saturating GPU utilization using MPI and CUDA Multi-Process Service, refactoring code to leverage batched linear algebra operations on GPU, and interleaving IO with computation.
- **25x** improvement in per-node throughput using Perlmutter compared to Edison baseline.



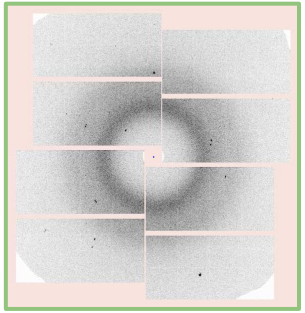
ExaFEL

XFEL requires **real-time data analysis** to make decisions **during ongoing experiments**. Data collection rates outpacing computational resources at the experimental sites, **requiring a Superfacility approach**.

In two years, NESAP has developed a highly scalable CUDA/GPU application.
CCTBX/nanoBragg w/ runtime improved from 12.3 hours on Edison, to 2 minutes



CCTBX/nanoBragg strong scaling on Summit. Colored lines show number of concurrent streams per GPU

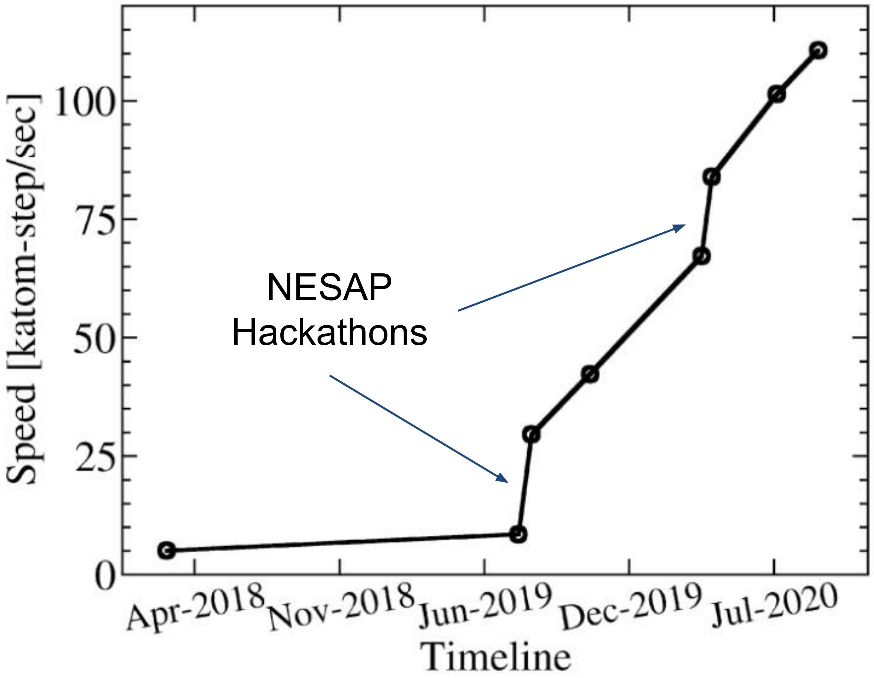


LAMMPs

- LAMMPS is a classical molecular dynamics code with a focus on materials modeling
- Production LAMMPS/Kokkos version was highly optimized over a serious of hackathons - Joint effort of NERSC/NESAP, ECP, NVIDIA and HPE
- Every kernel was rewritten and optimized individually, compared to baseline
- **22x** improvement in performance compared to baseline on NVIDIA V100 GPU (previous generation than on Perlmutter).
- SSI is the system-wide throughput increase over Edison in atom-steps/second.

SSI: 69

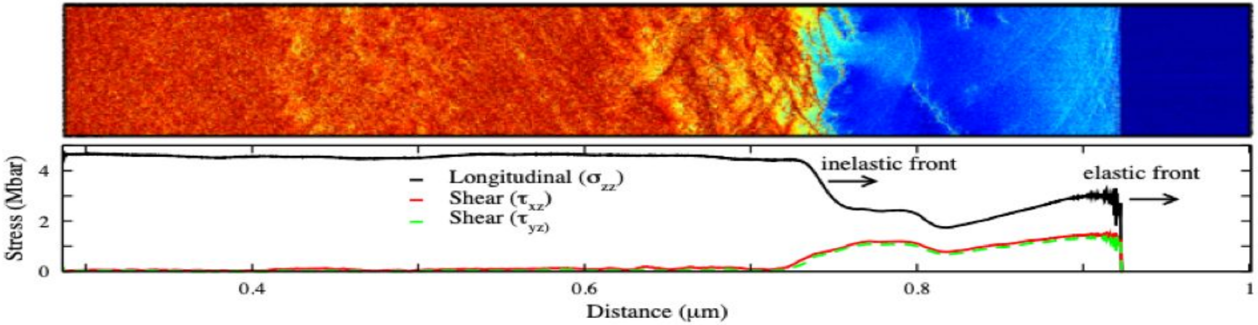
Node vs Node Speedup: 250x



NESAP Hackathons

Record Scale MD With LAMMPs Gordon Bell Finalists

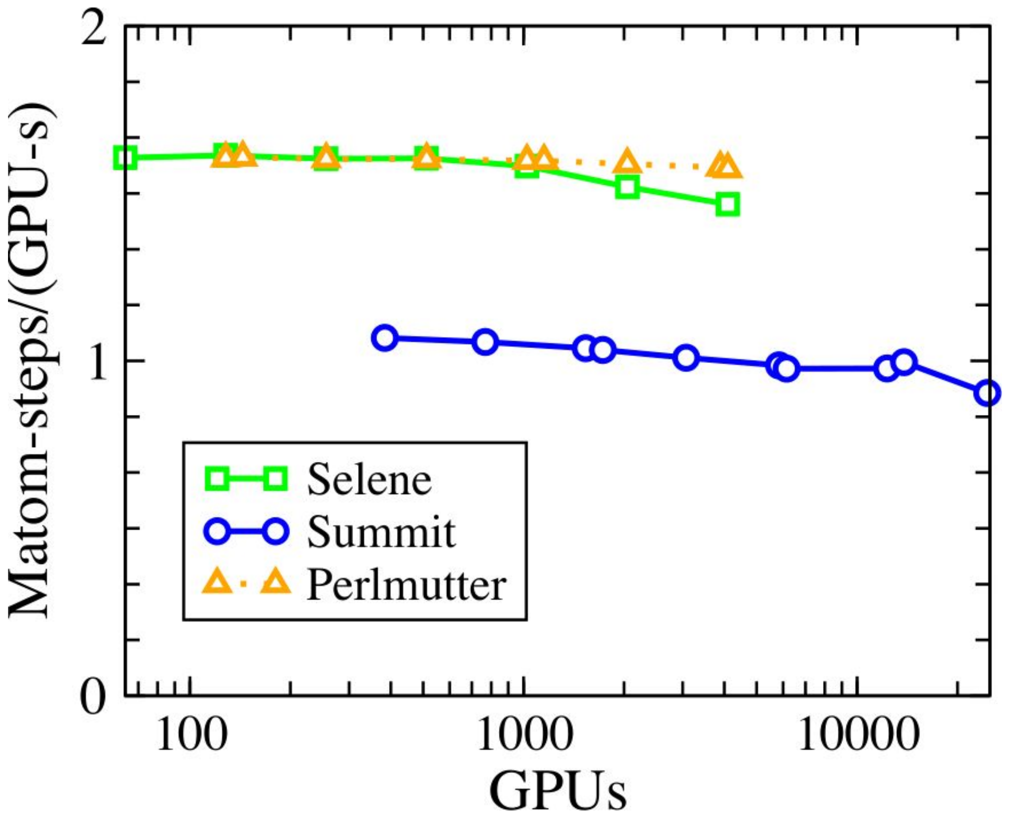
- Collaborative effort: University of South Florida, Sandia, NERSC and NVIDIA
- Billion atom molecular dynamics simulation (20B atoms)
  - SNAP quantum-accurate machine learned interatomic potential
  - Kokkos CUDA backend for NVIDIA GPUs
  - A run achieved 11.24 PFLOPS on Perlmutter on 1024 nodes (~ 2/3rd of the total machine)
- Simulation model shock compression of carbon at extreme pressures and temperatures.



1.8 billion carbon atom simulation of split elastic-inelastic shock wave propagating in single crystal diamond (dark blue). The elastic precursor (light blue) is followed by an inelastic wave (red), which exhibits an unexpected stress relaxation mechanism

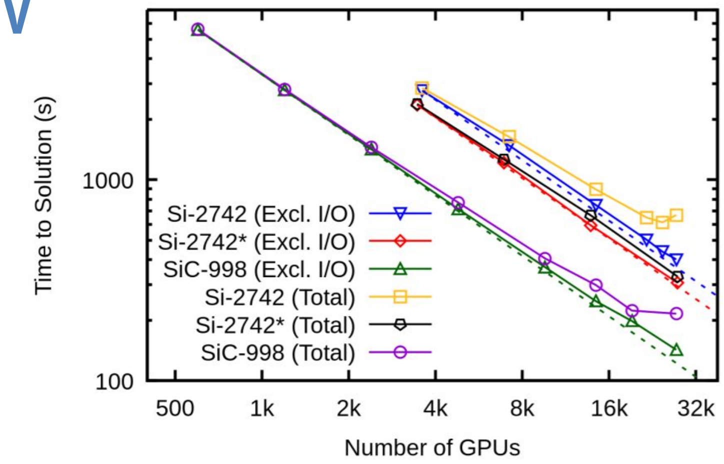
Record Scale MD With LAMMPs Gordon Bell Finalists

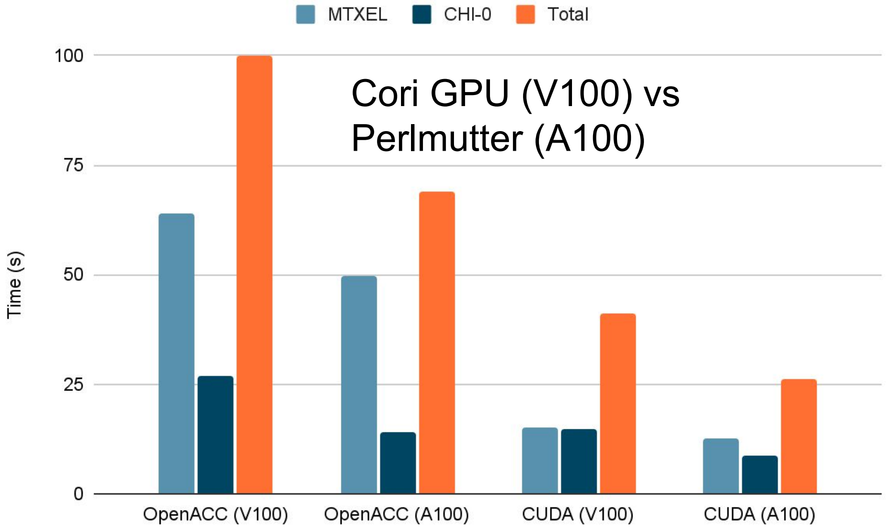
Strong scaling the amorphous carbon problem on Perlmutter and related systems.



Qubit Design w/ BerkeleyGW

The BerkeleyGW NESAP team was recognized as a Gordon Bell finalist in 2020.



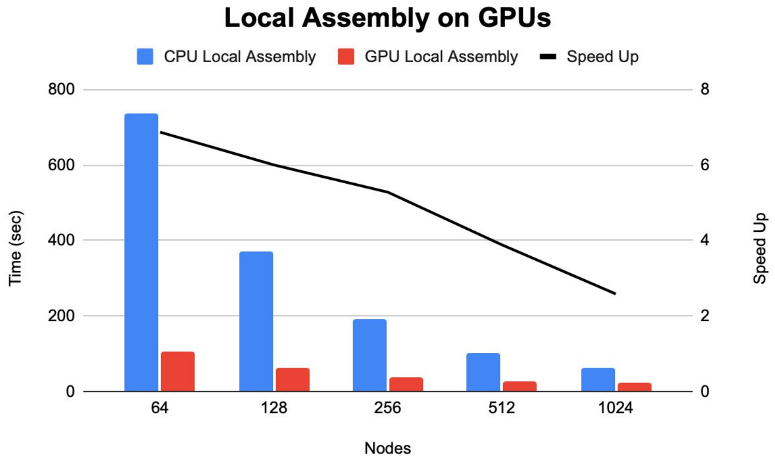

Cori GPU (V100) vs Perlmutter (A100)

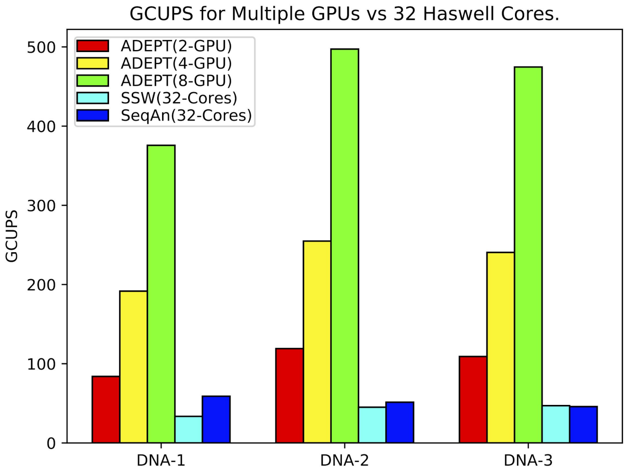
	MTXEL	CHI-0	Total
OpenACC (V100)	64	27	100
OpenACC (A100)	49.8	14.2	69
CUDA (V100)	15.2	14.7	41
CUDA (A100)	12.6	8.7	26.2

- Si-214 system (scaled: 4Ry CT ; 3000 bands). 8 GPUs each.

Exabiome (Meta-Genomics)

- A lot of progress has been made on GPU algorithms for meta-genomics.
- This NESAP team wrote the world’s fastest GPU aligners using a lot of clever strategies, newly available GPU intrinsic instructions etc.
- With the help of warp level intrinsics, dynamic data structures were written for GPUs from scratch to re-write the Local Assembly stage.

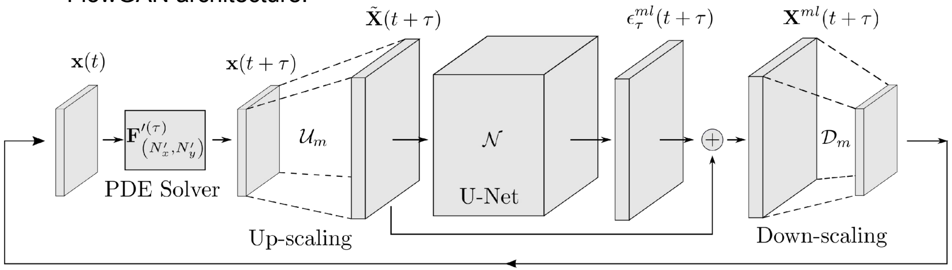




Accelerating CFD with GANs on Perlmutter

The FlowGAN project introduces a technique based on a deep neural network architecture to augment traditional numerical simulations of fluid flows. The ML model is used to correct the numerical errors induced by a coarse-grid simulation of turbulent flows at high-Reynolds numbers.

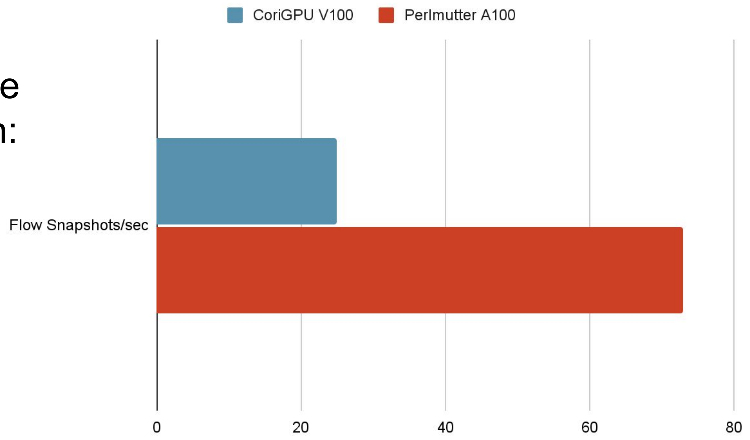
FlowGAN architecture:



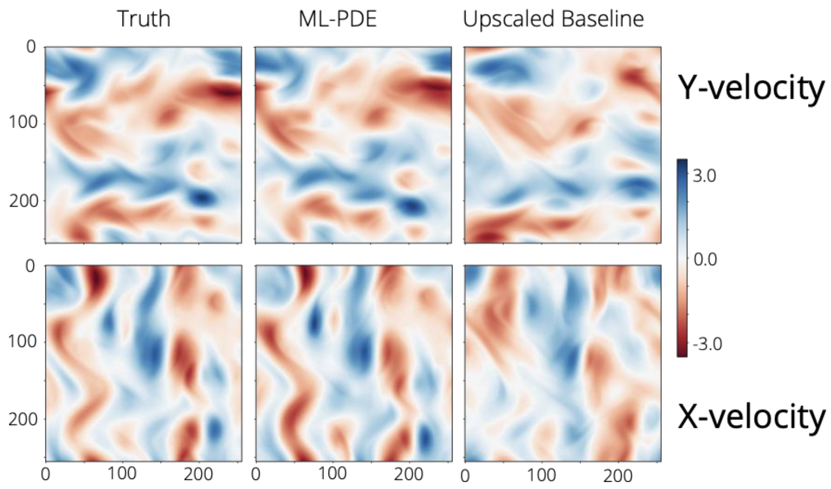
$\mathbf{x}(t)$
$\mathbf{F}'^{(\tau)}_{(N'_x,N'_y)}$
PDE Solver
$\mathbf{x}(t+\tau)$
$\mathcal{U}_m$
Up-scaling
$\tilde{\mathbf{X}}(t+\tau)$
$\mathcal{N}$
U-Net
$\epsilon^{ml}_{\tau}(t+\tau)$
$\mathbf{X}^{ml}(t+\tau)$
$\mathcal{D}_m$
Down-scaling

Performance Comparison:

2.9x performance improvement over CoriGPU on ML workflow



$t = 10$ model time units



Key Takeaways

- NERSC successful in preparing a significant number of key Office of Science applications for Perlmutter. Keys to success:
  - Early engagement and access to GPU technologies
  - Embedded Postdocs
  - Focused Hackathons
- NERSC continuing to engage w/ broad community to enable use of Perlmutter productively
  - Encouraging community to join GPUHackathons.org events all over the country
- GPU optimizations (Increasing Parallelism, Understanding and Minimizing Code Movement) continue on themes from Cori
- OpenMP and C++ Frameworks (Kokkos etc.) are viable performance portable options.

Thank you !