HPC Resources at NERSC

Brian Friesen
Lawrence Berkeley National Laboratory
August 14, 2018

What I assume about you

- You would like to be scientifically productive using BOUT++ with your computing allocation at NERSC
- You would like to (continue to) improve BOUT++ to achieve a large fraction of peak performance on NERSC resources, and to maximize weak- and strong-scaling efficiency

Objectives today

- Demonstrate that you can be efficient and productive using NERSC HPC resources (SW and HW) to perform BOUT++ simulations
- Introduce performance engineering tools available at NERSC which guide optimization at different scales
  - Node level: maximize usage of available hardware
  - System level: improving load balance, I/O, minimizing communication

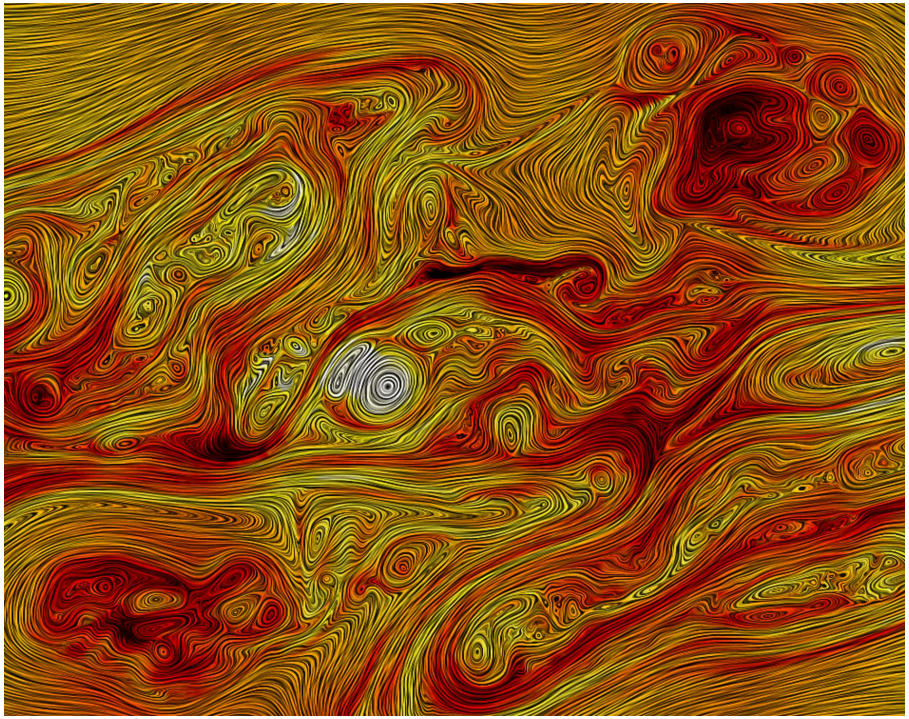
Part 1: Production Resources

About NERSC

- National Energy Research Scientific Computing Center
  - Established 1974 as CTRCC @ LLNL
  - first unclassified supercomputer center in the USA
  - Original mission: to enable computational science as complement to magnetically controlled plasma experiment
- Today's mission: ***Accelerate scientific discovery at the DOE Office of Science through High-Performance Computing and Extreme Data Analysis***
- NERSC is a national user facility

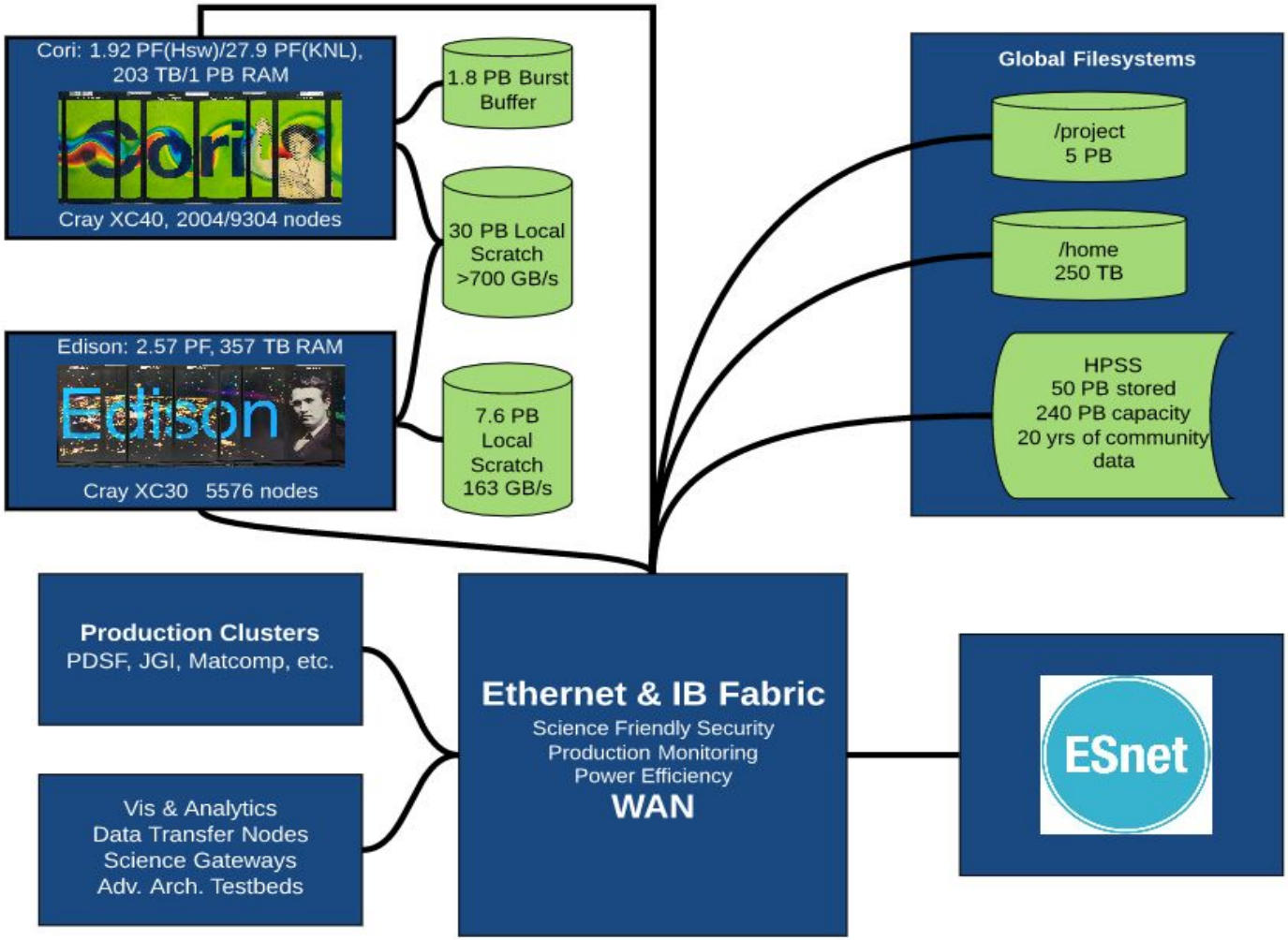
About NERSC

- Diverse workload:
  - 7000 users, 800 projects
  - 600 codes, 100s of users daily
- Allocations primarily controlled by DOE
  - 80% DOE Annual production awards (ERCAP)
    - O(10K)-O(10M) hour awards
    - Proposal-based, chosen by DOE program managers
  - 10% DOE ASCR Leadership Computing Challenge
  - 10% NERSC reserve



Turbulence in Solar Wind

NERSC Systems Map



HPC Systems

Edison:

- Cray XC30 - 2.5 PF peak
- 5576 nodes of Intel Xeon E5-2695 v2 @ 2.40GHz ("Ivy Bridge")
- 12 cores/socket, 2 sockets/node

Cori:

- Cray XC40 - 30 PF peak
- 2388 nodes of Intel Xeon E5-2698 v3 @ 2.3 GHz ("Haswell")
  - 16 cores/socket, 2 sockets/node
- 9688 nodes of Intel Xeon Phi 7250 ("Knights Landing")
  - 68 cores/socket, 1 socket/node
- 1.5 PB SSD-based "burst buffer" for high-performance I/O

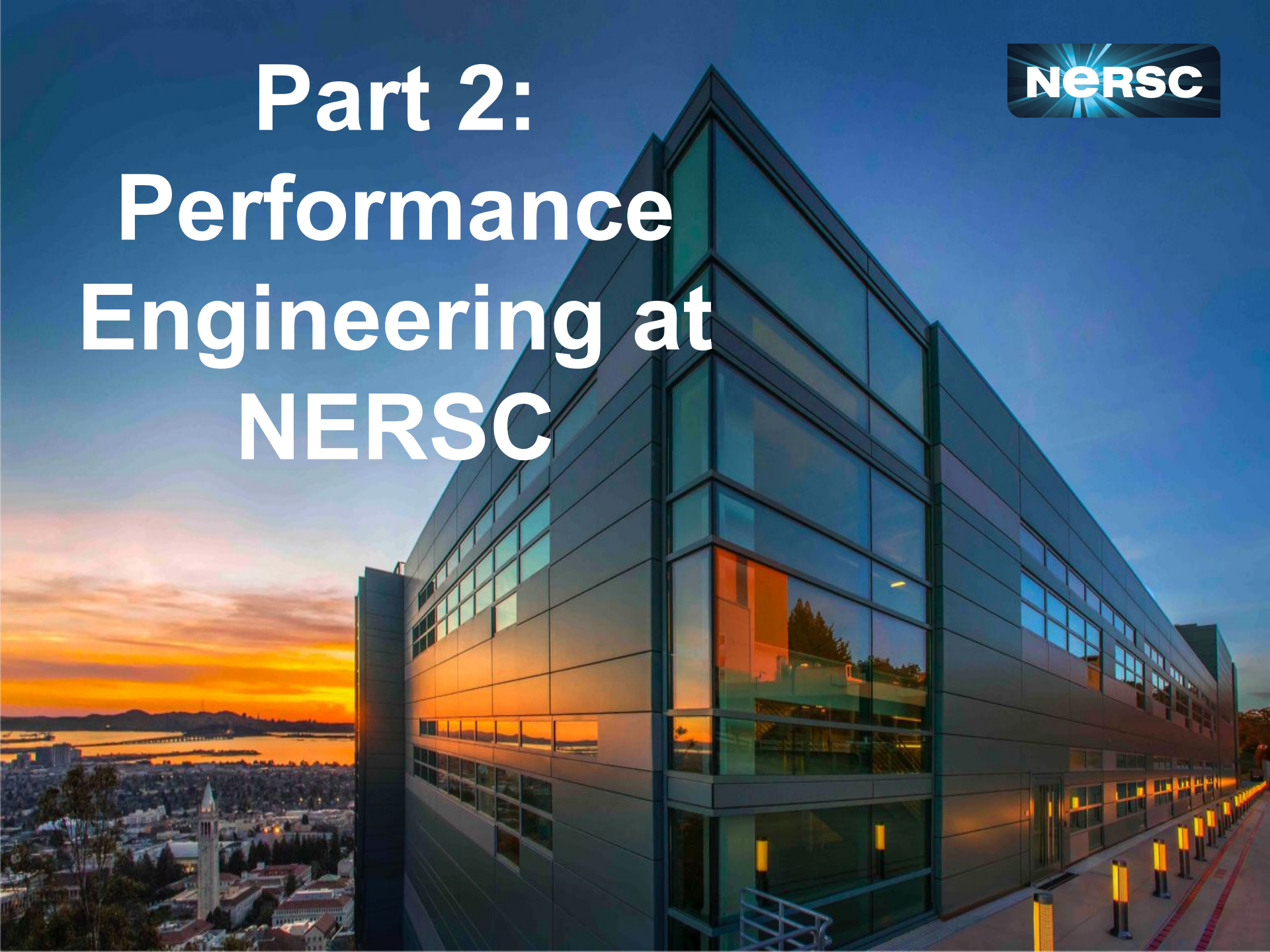
HPC System Software

- 3 “programming environments”: **GCC**, **Intel**, **Cray**
  - Support code-generation for IVB, HSW, KNL
- **Cray MPI** (MPICH-based) optimized for Aries HSN
- **Cray LibSci**: optimized BLAS, (Sca)LAPACK, FFTW, PETSc, Trilinos, IRT
- **Cray TPSL**: MUMPS, SuperLU(_DIST), (Par)METIS, HYPRE, SUNDIALS, Scotch, matio, GLM
- *Lots* of performance engineering and debugging software: CrayPat, Intel VTune, Intel Advisor, Arm MAP/DDT, TotalView, gdb4hpc

Filesystems

- Global Filesystems:
  - Home (40 GB per user)
  - Project (varies)
- Local Filesystems:
  - Scratch (20 TB per user; varies)
  - Burst Buffer (no quotas)
- Long-term Storage System:
  - HPSS (many TB per user)

Part 2: Performance Engineering at NERSC



Scaling “up”

- My favorite tools for single-node performance analysis are Intel VTune and Advisor
- Both provide highly detailed description of performance characteristics and bottlenecks on a single compute node
  - Are all OpenMP threads busy?
  - Do performance-critical loops vectorize efficiently?
  - Does the code have good cache reuse?

Scaling “up”

NoMachine - NX5Configure

/global/cscratch1/sd/friesen/BOUT/advisor1 - Intel Advisor <@cori12>

File View Help

Start Survey Analysis

Welcome | e000

Elapsed time: 38.85s | Vectorized | Not Vectorized | Smart Mode OFF

FILTER: All Modules | All Sources | Loops And Functions | All Threads

INTEL ADVISOR 2018

Summary | Survey & Roofline | Refinement Reports

ROOFLINE

Function Call Sites and Loops	Performance Issues	Self Time	Total Time	Type
calc_index		4.464s	4.464s	Inlined Function
[libfftw3.so.mpi31.3]		3.524s	3.524s	Function
VDDX_WENO3		1.849s	1.849s	Function
[loop in Mesh::applyXdiff at index_derivs.cxx:845]	**2 User function call(s) present**	**1.312s**	**2.643s**	**Scalar**
Laplacian::tridagCoefs		0.881s	1.270s	Function
[loop in operator/ at field3d.cxx:864]	1 Unoptimized floating point operation processing possible	0.806s	1.427s	Scalar
[loop in pvode::N_VDivompparallel_for@323 at nvec ...	2 Unoptimized floating point operation processing possible	0.728s	0.728s	Vectorized+Threa ...

Source | Top Down | Code Analytics | Assembly | Recommendations | Why No Vectorization?

File: index_derivs.cxx:845 Mesh::applyXdiff

Line	Source	Total Time	%	Loop/Function Time	%	Traits
843	// More than one guard cell, so set pp and mm values					
844	// This allows higher-order methods to be used					
845	for(const auto &i : result.region(region)) {	0.249s		2.643s		
	[loop in Mesh::applyXdiff at index_derivs.cxx:845] Scalar loop. Not vectorized: exception handling for a call prevents vectorization No loop transformations applied					
846	stencil s;	0.290s				
	Selected (Total Time):	0.249s				

/global/cscratch1/sd/friesen/BOUT/advis | cori :

Scaling "up"

NoMachine - NX5Configure

<no current project> - Intel VTune Amplifier <@cori12>

Welcome | vtune3

Advanced Hotspots Hotspots viewpoint (change)

INTEL VTUNE AMPLIFIER 2018

Analysis Target | Analysis Type | Collection Log | Summary | Bottom-up | Caller/Callee | Top-down Tree | Platform

Grouping: Function / Call Stack

Function / Call Stack	CPU Time	Instructions Retired	CPI Rate	CPU Frequency Ratio	Module	Function (Full)
_INTERNAL_25_____src_kmp_barrier_cpp_fa6086	48.030s	42,714,000,000	1.685	1.070	libiomp5.so	void _INTERNAL_25_____src_kmp_barrier_cpp_fa60
[vmlinux]	8.870s	2,618,000,000	5.251	1.107	vmlinux	[vmlinux]
__kmp_wait_yield_4	3.690s	2,184,000,000	2.500	1.057	libiomp5.so	__kmp_wait_yield_4
VDDX_WENO3	1.270s	476,000,000	3.912	1.047	blob2d	VDDX_WENO3(double, stencil&)
__kmp_yield	0.840s	56,000,000	24.750	1.179	libiomp5.so	__kmp_yield
calc_index	0.840s	1,302,000,000	1.032	1.143	blob2d	calc_index(bindex*)
kmp_basic_flag_native<unsigned long long>::notdone	0.680s	42,000,000	27.333	1.206	libiomp5.so	kmp_basic_flag_native<unsigned long long>::notdone_ch
_INTERNAL_25_____src_kmp_barrier_cpp_fa6086	0.560s	0		0.982	libiomp5.so	_INTERNAL_25_____src_kmp_barrier_cpp_fa608613
sched_yield	0.430s	14,000,000	41.000	0.953	libc-2.22.so	sched_yield
Field3D::operator()	0.370s	350,000,000	0.280	0.189	blob2d	Field3D::operator()(int, int, int)
Laplacian::tridagCoefs	0.370s	406,000,000	1.655	1.297	blob2d	Laplacian::tridagCoefs(int, int, double, std::complex<doub
Mesh::applyXdiff	0.270s	364,000,000	1.500	1.444	blob2d	Mesh::applyXdiff(Field3D const&, double (*)(stencil&), Cl
imprt	0.250s	322,000,000	0.913	0.840	libfftw3.so.mpi31.3.5.6	imprt

0s 1s 2s 3s 4s 5s 6s 7s 8s 9s 10s

Thread: OMP Master Thread #0 (TID... | OMP Worker Thread #3 (TID... | OMP Worker Thread #2 (TID... | OMP Worker Thread #4 (TID... | OMP Worker Thread #5 (TID... | OMP Worker Thread #6 (TID...

CPU Time

Ruler Area: Region Instance | OpenMP Barrier-to-Barrier Segment

Thread: Running | CPU Time | Spin and Overhead ... | CPU_CLK_UNHAL...

FILTER 100.0% | Any Process | Thread Any Thread | Module Any Module | Any Utilizatio | User/system functions | Show inline functic | Functions only

<no current project> - Intel VTune Ampli | cori :

Scaling “up”

- Limitations of VTune/Advisor
  - Difficult to use when running on > 1 compute nodes
  - Data collection rate is <u>very</u> high, and user has little control over it
    - Effectively limits window of execution time which can be profiled (even ~1 min on 1 node is pushing it)
    - Clunky CLI with lots of annoying “gotchas”

Scaling "out"

- My favorite tool for multi-node performance analysis is CrayPat
- Works with any PrgEnv that Cray supports (Intel, GCC, CCE)
- Low overhead, even at high concurrency
  - ~10% at 64 Ki MPI ranks for sampling
- Easy to use, has "reasonable set of defaults", no annoying "gotchas"
- Works with MPI, OpenMP, CUDA, UPC, coarrays, SHMEM

Scalir

```
Table 1:  Profile by Function

  Samp% |      Samp |  Imb. |  Imb. | Group
        |           |  Samp | Samp% |  Function
        |           |       |       |   Thread=HIDE

 100.0% | 23,382.0 |    -- |    -- | Total
|-----------------------------------------------------------
|  94.1% | 22,006.0 |    -- |    -- | USER
||----------------------------------------------------------
||  15.9% |  3,725.0 |    -- |    -- | VDDX_WENO3
||  13.5% |  3,168.0 |    -- |    -- | next_index3
||  10.2% |  2,396.0 |    -- |    -- | operator/
||   7.2% |  1,677.0 |    -- |    -- | operator*
||   6.5% |  1,523.0 |    -- |    -- | tridag
||   5.8% |  1,362.0 |    -- |    -- | Laplacian::tridagCoefs
||   5.7% |  1,343.0 |    -- |    -- | Field3D::operator[] const
||   4.5% |  1,055.0 |    -- |    -- | operator+
||   3.7% |    868.0 |    -- |    -- | Mesh::applyXdiff
||   3.3% |    771.0 |    -- |    -- | Solver::loop_vars_op
||   2.5% |    591.0 |    -- |    -- | Field3D::setXStencil const
||   2.3% |    530.0 |    -- |    -- | Field3D::setZStencil const
||   1.7% |    404.0 |    -- |    -- | Coordinates::Delp2
||   1.5% |    340.0 |    -- |    -- | sliceXZ
||   1.4% |    321.0 |    -- |    -- | Mesh::indexVDDX
||   1.1% |    265.0 |    -- |    -- | Mesh::indexVDDZ
||   1.1% |    265.0 |    -- |    -- | DataIterator::offset const
```

```
 100.0% | 23,382.0 | Total
|-----------------------------------------------------------------------------
|  94.1% | 22,006.0 | USER
||----------------------------------------------------------------------------
||  15.9% |  3,725.0 | VDDX_WENO3
|||---------------------------------------------------------------------------
3||   7.9% |  1,840.0 | Mesh::indexVDDX:index_derivs.cxx:line.1833
4||        |          |  VDDX:derivs.cxx:line.362
5||        |          |   bracket:difops.cxx:line.966
||||||-------------------------------------------------------------------------
6|||||   4.0% |    928.0 | Blob2D::rhs:blob2d.cxx:line.131
7|||||        |          |  Solver::run_rhs:solver.cxx:line.1271
8|||||   3.9% |    918.0 |   solver_f:pvode.cxx:line.299
9|||||   2.7% |    628.0 |    _INTERNALbb55bf6f::pvode::CVSpgmrAtimesDQ:cvspgmr.cpp:line.466
10||||        |          |     pvode::SpgmrSolve:spgmr.cpp:line.275
11||||        |          |      _INTERNALbb55bf6f::pvode::CVSpgmrSolve:cvspgmr.cpp:line.390
12||||        |          |       _INTERNAL4d727760::pvode::CVStep:cvode.cpp:line.1985
13||||        |          |        pvode::CVode:cvode.cpp:line.886
14||||        |          |         PvodeSolver::run:pvode.cxx:line.244
15||||        |          |          Solver::solve:solver.cxx:line.559
16||||        |          |           main:blob2d.cxx:line.161
6|||||   3.9% |    912.0 | Blob2D::rhs:blob2d.cxx:line.146
7|||||        |          |  Solver::run_rhs:solver.cxx:line.1271
8|||||   3.9% |    905.0 |   solver_f:pvode.cxx:line.299
9|||||   2.7% |    627.0 |    _INTERNALbb55bf6f::pvode::CVSpgmrAtimesDQ:cvspgmr.cpp:line.466
10||||        |          |     pvode::SpgmrSolve:spgmr.cpp:line.275
```

Scaling “out”

- Limitations of CrayPat
  - Less information than VTune about hardware performance metrics
  - It can read most of the same HWPCs but you have to figure out the magic formulas that the Intel tools have
  - Requires program instrumentation (CrayPat modules must be active at compile-time)
  - Only available on Cray systems

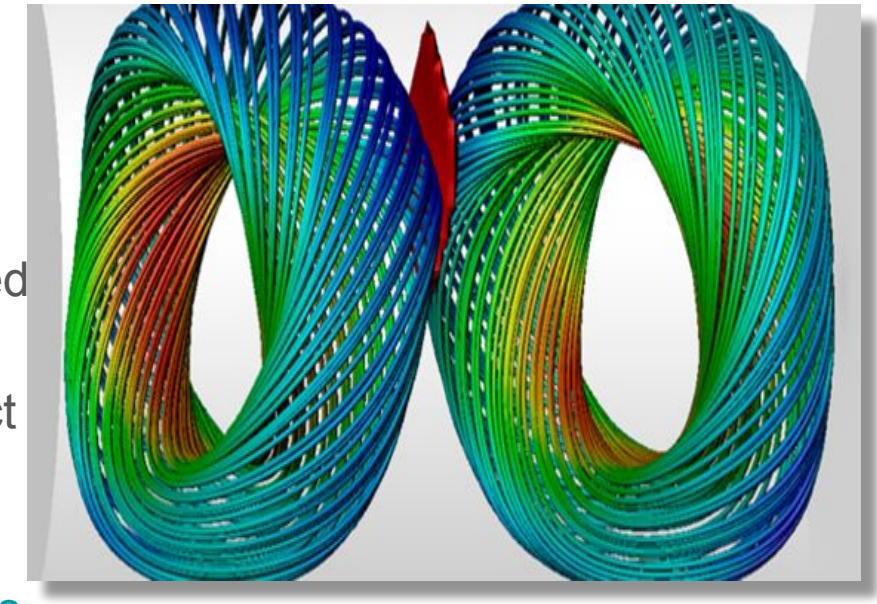
Epilogue

NERSC User Group (NUG)

- Community of NERSC users
- Source of advice and feedback for NERSC (we listen!)
- Executive Committee: 3 representatives from each office + 3 members-at-large
- Monthly teleconferences hosted by NERSC (usually 3rd Thursday of the month, 11 am to noon)

Help NERSC Help You!

- Be sure to acknowledge NERSC in publications!
  - This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.
  - Available at: https://www.nersc.gov/users/accounts/user-accounts/acknowledge-nersc/
- Science highlights sent to DOE each quarter
  - Please send us links to your publications!



Magnetic field lines from HiFi simulations of two spheromaks. NERSC repo m1255
Image courtesy of Vyacheslav Lukin (NRL)

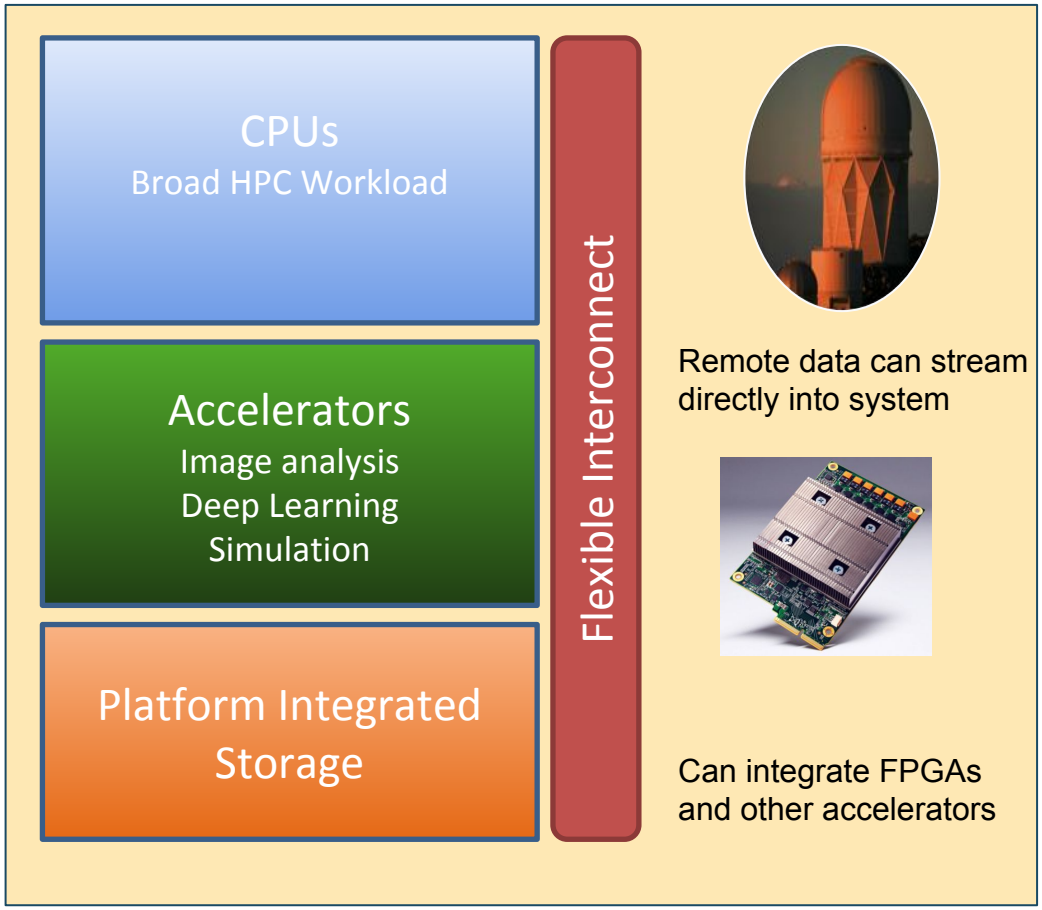
NERSC-9: A 2020 Pre-Exascale Machine

Capabilities

- **3-4x capability of Cori**
- **Optimized for both simulation and data analysis**
- **Looks ahead to exascale with specialization and heterogeneity**

Features

- **Energy efficient architecture**
  - Large amount of high-BW memory
  - High-BW, low-latency network
- **Production deployment of accelerators for the DOE community**
- **Single-tier all-flash HPC filesystem**



System will be announced in 2018

Useful resources

- Slurm job script generator for NERSC systems
  - https://my.nersc.gov/script_generator.php
- Application porting and performance
  - http://www.nersc.gov/users/computational-systems/cori/application-porting-and-performance/
- Using the burst buffer on Cori
  - http://www.nersc.gov/users/computational-systems/cori/burst-buffer/
- Using Intel Advisor at NERSC
  - http://www.nersc.gov/users/software/performance-and-debugging-tools/advisor/
- Using Intel VTune at NERSC
  - http://www.nersc.gov/users/software/performance-and-debugging-tools/vtune/
- Using CrayPat at NERSC
  - http://www.nersc.gov/users/software/performance-and-debugging-tools/craypat/
- CFP: “High Impact Science at Scale on Cori”
  - http://www.nersc.gov/users/announcements/featured-announcements/high-impact-science-at-scale-2/