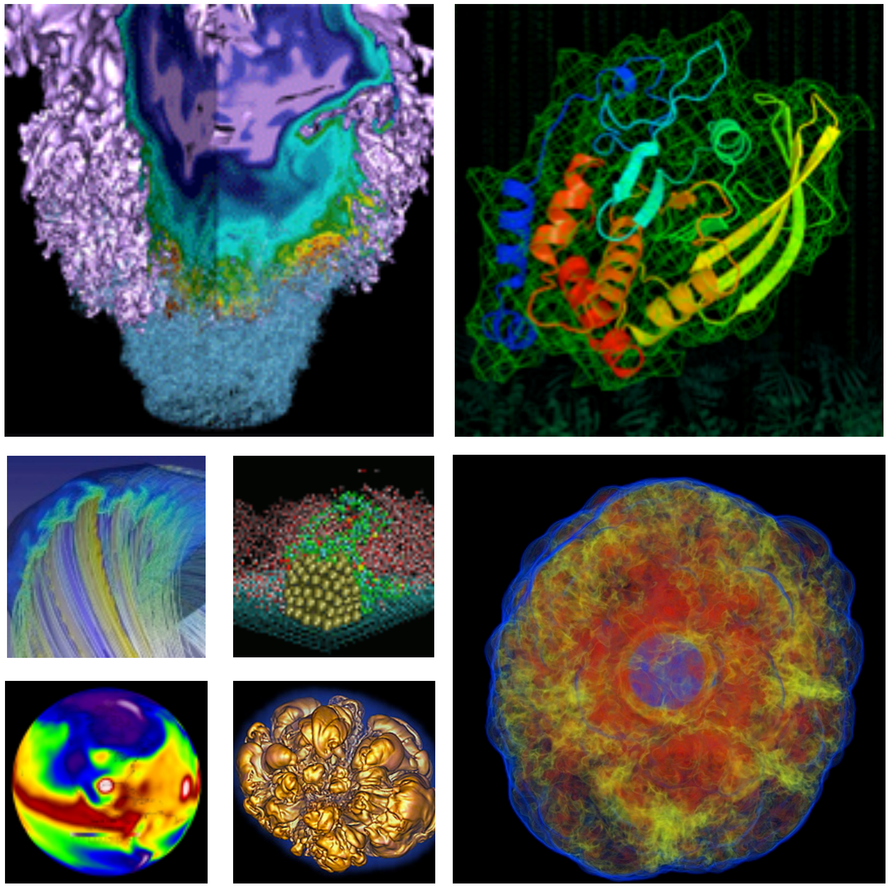
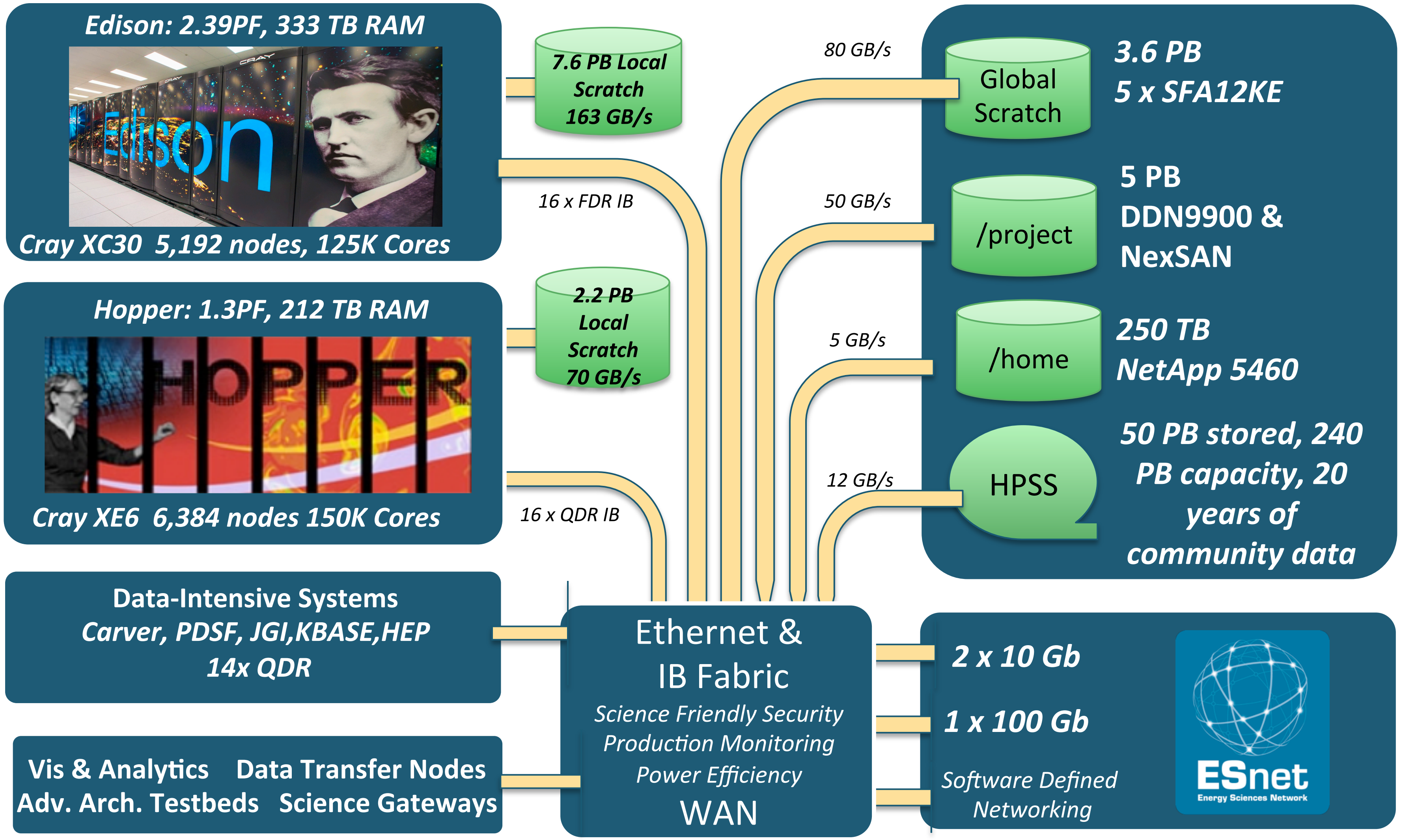
NERSC Systems Update



Helen He

NERSC Climate PIs Telecon
Dec 4, 2015

NERSC Systems Early 2015



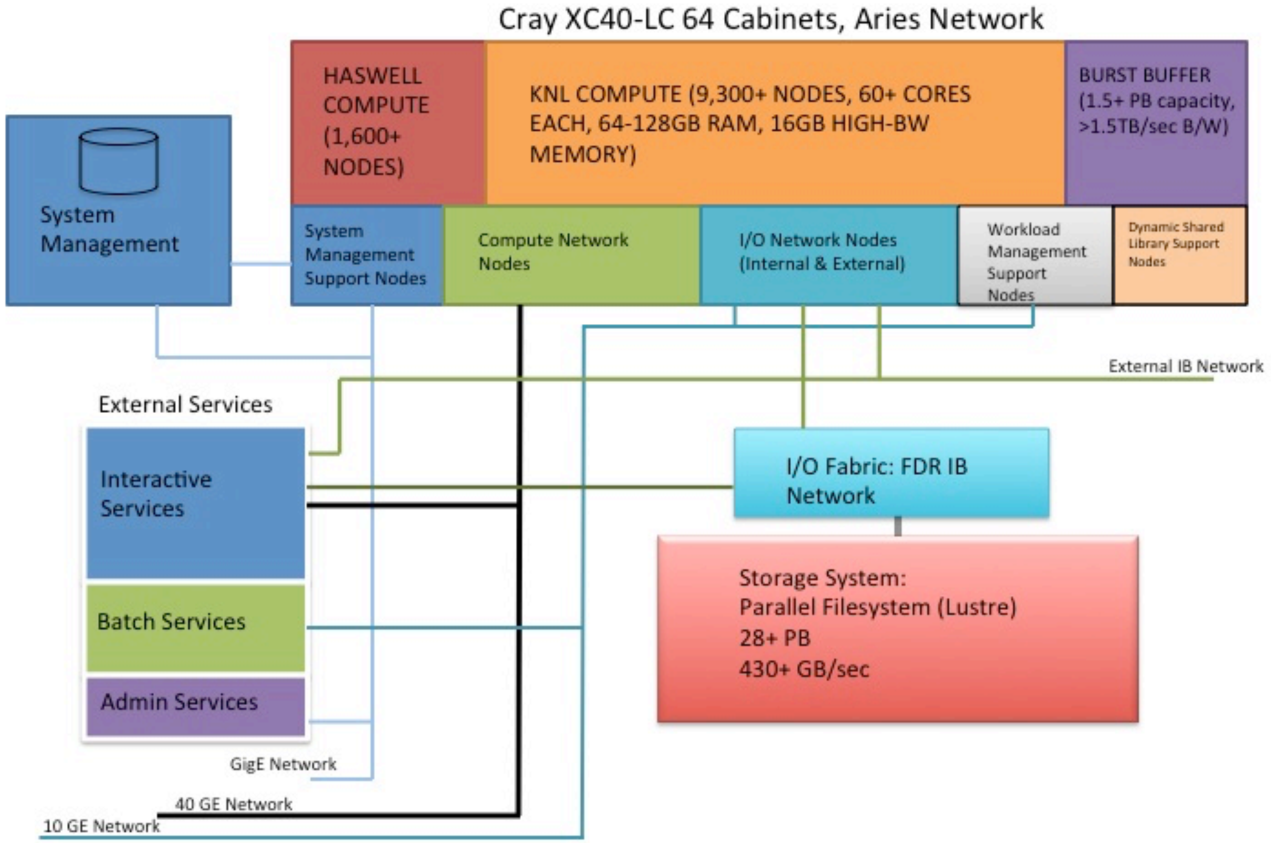
Impact of NERSC Move and System Retirement

- **Hopper retirement**
  - Noon, Dec 15. All jobs either running and in the queue will be killed.
  - Login nodes and /scratch file systems available until Dec 22 or earlier if there are any issues. No recovery effort due to beyond contract date.
- **Global file systems migrate to CRT**
  - /global/project: 11/30 - 12/13
  - /global/homes, /global/common, /global/syscom: 12/14 - 12/16
  - No outage. During the migration, users may observe a reduction of file system performance.
- **Edison was powered off and moved to CRT at Nov 30, expect to be offline for up to 6 weeks.**

Cori Configuration (scheduled mid 2016)

- **64 Cabinets of Cray XC System**
  - Over 9,300 ‘Knights Landing’ (KNL) compute nodes
    - Self-hosted (not an accelerator)
    - Greater than 60 cores per node with four hardware threads each
    - 64-128 GB memory per node
    - High bandwidth on-package memory
  - 1,630 ‘Haswell’ compute nodes (Phase 1, delivered Aug 2015)
    - Data partition
  - 23 external login nodes
  - Aries Interconnect (same as on Edison)
  - >5x Edison application performance using NERSC SSP metric
- **Lustre File system**
  - 28 PB capacity, 432 GB/sec peak performance
- **NVRAM “Burst Buffer” for I/O acceleration**
  - ~1.5PB capacity, > 1.5 TB/sec I/O bandwidth
- **Significant Intel and Cray application transition support**
- **Delivery in two phases, summer 2015 and summer 2016. Installation in new LBNL CRT**

The Cori System

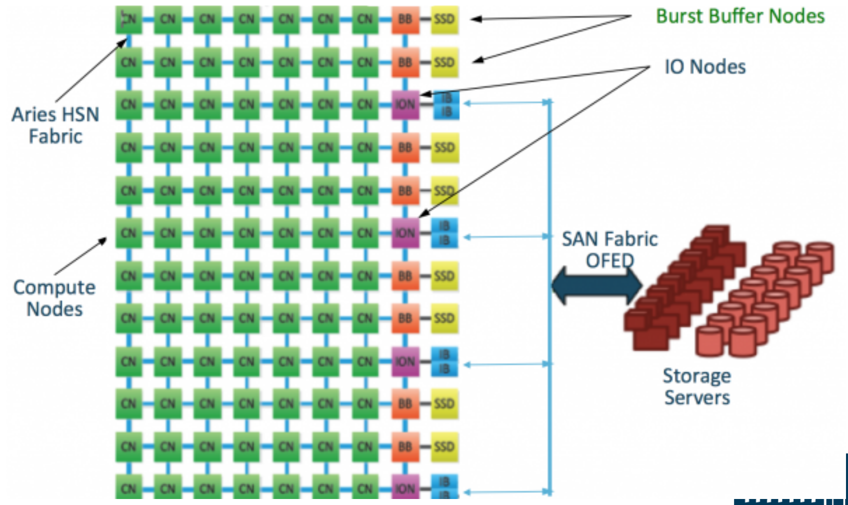

Cray XC40-LC 64 Cabinets, Aries Network
HASWELL COMPUTE (1,600+ NODES)
KNL COMPUTE (9,300+ NODES, 60+ CORES EACH, 64-128GB RAM, 16GB HIGH-BW MEMORY)
BURST BUFFER (1.5+ PB capacity, >1.5TB/sec B/W)
System Management
System Management Support Nodes
Compute Network Nodes
I/O Network Nodes (Internal & External)
Workload Management Support Nodes
Dynamic Shared Library Support Nodes
External IB Network
External Services
Interactive Services
Batch Services
Admin Services
I/O Fabric: FDR IB Network
Storage System: Parallel Filesystem (Lustre) 28+ PB 430+ GB/sec
GigE Network
40 GE Network
10 GE Network

Cori Phase 1 (Data Partition)

- **Installed at CRT in late Aug 2015.**
- **Early Users Program in phases from mid Oct to early Nov.**
- **All users enabled on Nov 11.**
- **Data partition: 1,630 "Haswell" compute nodes**
  - Large memory nodes and throughput optimized processors: 128 GB/node
  - Burst buffer – NVRAM flash nodes on the interconnect fabric for IO caching: ~750 GB/sec I/O performance and ~ 750TB of storage. (Early User Program)
  - Larger Lustre file system: 30 PB of disk. > 700 GB/second I/O bandwidth. Ultimately to be mounted across NERSC systems as new global scratch.
  - Increased number of login/interactive nodes (>12) to support advanced workflows
- **SLURM workload manager**

Burst Buffer

- **The Burst Buffer on Cori is a layer of non-volatile storage that sits between the a processors' memory and the parallel file system.**
  - Improves application reliability (through faster checkpoint-restart)
  - Accelerates application I/O performance for small blocksize transfers and analysis files
  - Provides fast temporary space to out-of-core applications
  - Creates a staging area for jobs requiring large input files or persistent fast storage between coupled simulations
  - Post-processing analysis of large simulation data
  - In-transit visualization and analysis



A single system offers significant benefits to science

- **NERSC has two major strategic thrusts**
  - Useable Exascale
  - Enabling Data Intensive Computing
- **We currently deploy separate systems for compute intensive and data intensive workloads.**
- **Cori plus the Cori Phase 1 Data Partition will enable NERSC to make an impact on both**
- **Success depends on enabling the Cori Data Partition to meet the needs of Data Intensive Users**

Goals are to enable the analysis of large experimental data sets and in-situ analysis coupled to Petascale simulations

Running Jobs on Cori Phase 1

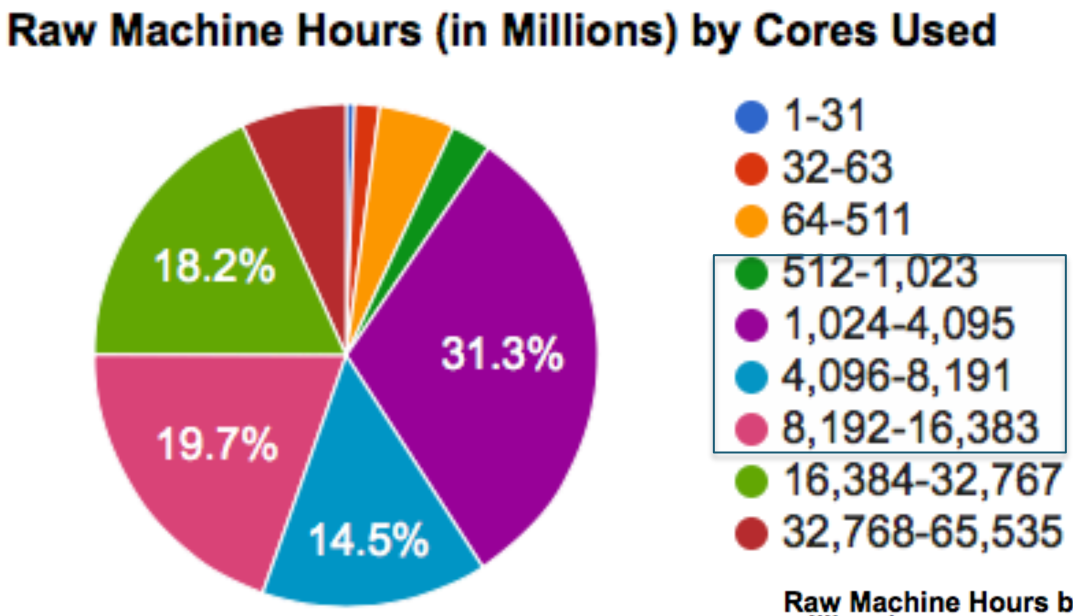
- **Very similar to running jobs on Edison, except for the processor, memory, and batch scheduler.**
  - Cori P1: 32 cores/node, 2 sockets/node, 16-core Haswell processor/socket at 2.3 GHz. 128 GB of memory per node.
  - Edison: 24 cores/node, 2 sockets/node, 12-core IvyBridge processor/socket at 2.4 HGz. 64 GB of memory per node.
  - Cori P1 uses SLURM batch scheduler (Edison used Torque/Moab before moving, will use SLURM when back online at CRT).
  - Refer to Cori Running jobs web page:
    - https://www.nersc.gov/users/computational-systems/cori/running-jobs/
- **Cori P1 delivers about the same computing power of Hopper. Free of charge before production.**

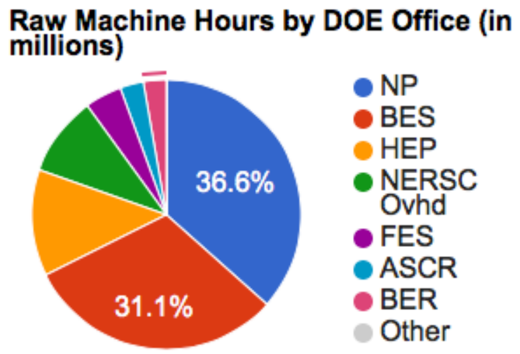
Advantages of Using SLURM

- **SLURM stands for Simple Linux Utility for Resource Management.**
- **SLURM unites the cluster resource management (such as Torque) and job scheduling (such as Moab) into one system. Avoids inter-tool complexity.**
- **Fully open source and extensible (plugin architecture)**
- **Low latency scheduling. Highly scalable.**
- **Integrated "serial" or "shared" queue**
- **Integrated Burst Buffer support**
- **Good memory management**
- **Built-in accounting and database support**
- **"Native" SLURM runs without Cray ALPS (Application Level Placement Scheduler)**
  - Batch script runs on the head compute node directly
  - Easier to use. Less chance for contention compared to shared MOM node.

Cori Phase 1 Machine Hours Delivered (10/02-12/02/15)

All users enabled on Nov 11





Raw Machine Hrs: Cori: 36,715,904 (100.0%)
Raw MPP Hrs*: Cori: 91,789,760 (100.0%)

Intel “Knights Landing” (KNL) Processor

- Next generation Xeon-Phi, >3TF peak
- Single socket processor - Self-hosted, not a co-processor, not an accelerator
- Up to 72 cores per processor with four hardware threads each
- Intel® "Silvermont" architecture enhanced for HPC
- Cores connected via a 2D mesh network
- Multiple NUMA domains per socket
- 512b vector units (32 flops/clock – AVX 512)
- 3X single-thread performance over current generation Xeon-Phi
- High bandwidth on-package memory, 16GB capacity with 5X bandwidth of DDR4 DRAM memory
- Higher performance per watt

Cori's Programming Environment

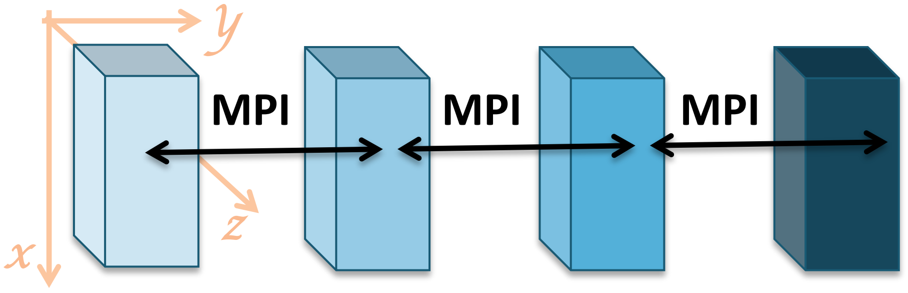
- **Key point: Cori will look basically just like Franklin/Hopper/ Edison to users**
- **Cori is <u>not</u> a heterogeneous or accelerator system; programmed via Fortran/C/C++ plus MPI+OpenMP**
- **Intel / Cray/ GNU programming environments and commitment from Cray and Intel for optimized math and I/O libraries**
- **Multiple vendor profiling tools: CrayPAT and Vtune**
  - Interest from 3rd-party suppliers, too
- **DDT and Totalview support + Cray debugging tools**

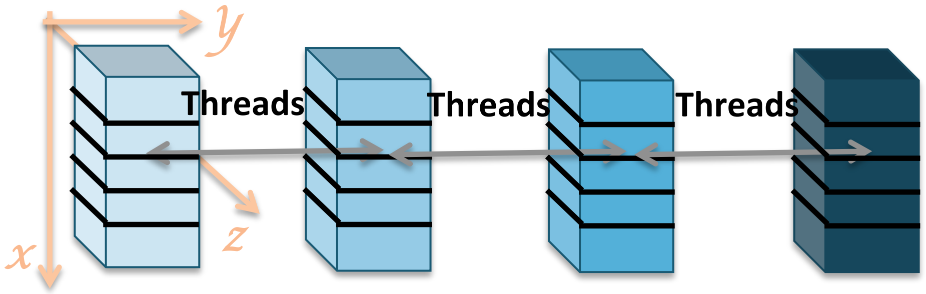
KNL Programming Model Considerations

- **Knight's Landing is a self-hosted part**
  - Users can focus on adding parallelism to their applications without concerning themselves with PCI-bus transfers
- **MPI + OpenMP preferred programming model**
  - Should enable NERSC users to make robust code changes.
  - Also helps to achieve scaling capability and code portability.
- **Why OpenMP?**
  - Expect between 1-2GB memory *per core*
  - With 2 threads/core memory/thread drops to less than 1 GB
  - Will need to use HW threads to get optimal performance on KNL
- **MPI-only will work – performance may not be optimal**
- **On package MCDRAM (High Bandwidth Memory)**
  - How to optimally use?
    - Cache (implicit), Flat (explicit) or Hybrid mode?

To run effectively on Cori users will have to:

- **Manage Domain Parallelism**
  - independent program units; explicit (MPI)
- **Increase Thread Parallelism**
  - independent execution units within the program; generally explicit (OpenMP)
- **Exploit Data Parallelism**
  - Same operation on multiple elements (vectorization)
- **Improve data locality**
  - Cache blocking; Use on-package memory (HBM)



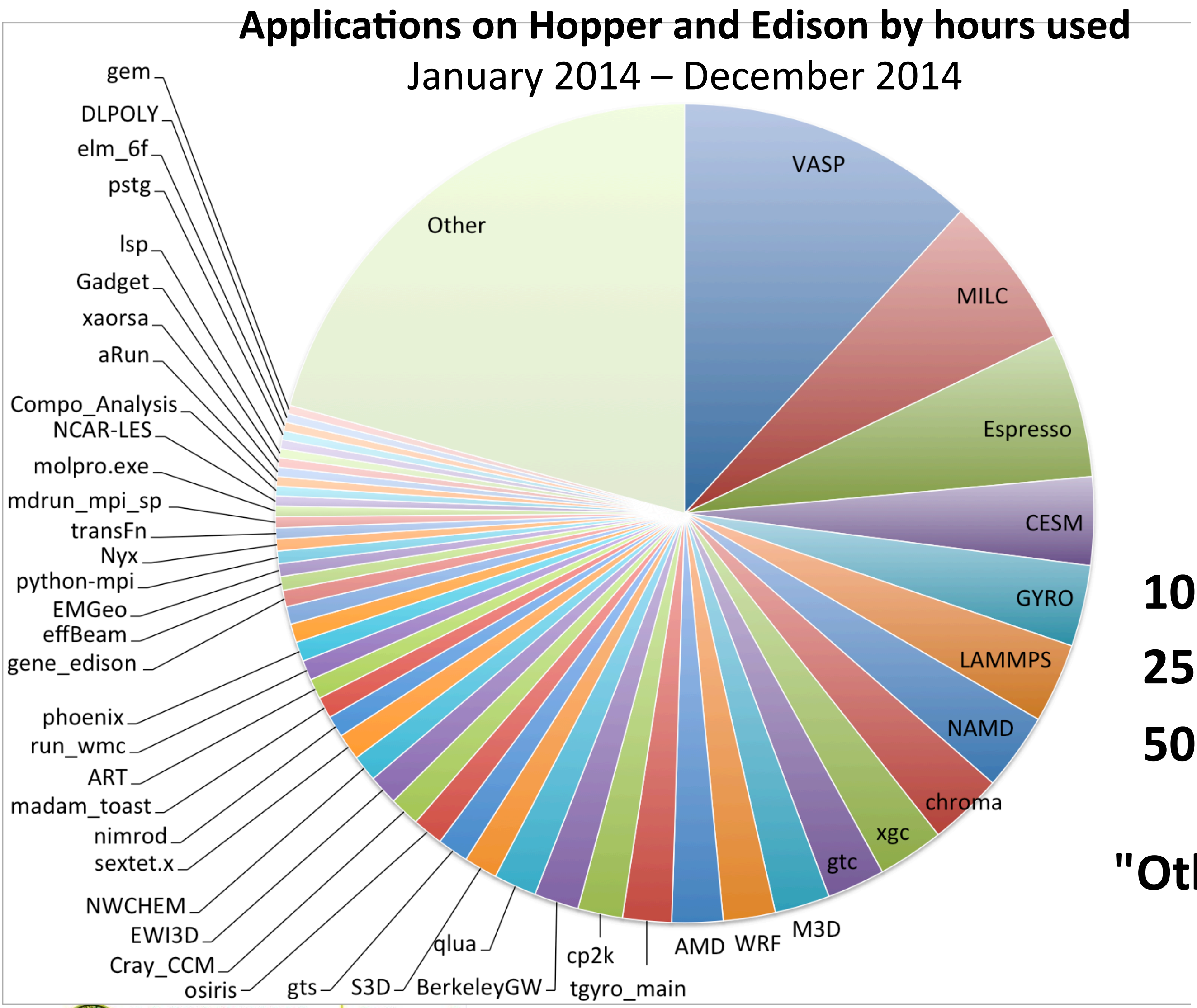


```
|--> DO I = 1, N
|         R(I) = B(I) + A(I)
|--> ENDDO
```

NERSC Exascale Science Application Program (NESAP)

- **Goal: to prepare DOE Office of Science user community for Cori manycore architecture**
- **20 applications were selected as Tier 1 (with postdocs) and Tier 2 applications to work closely with Cray, Intel and NERSC staff. Additional 26 Tier 3 teams. Share lessons learned with broader user community.**
- **Available resources are:**
  - Access to vendor resources and staff including “dungeon sessions” with Intel and Cray Center of Excellence
  - Early access to KNL “whitebox” systems
  - Early access and time on Cori
  - Trainings, workshops, and hackathons
  - Intel Xeon Phi User Group (IXPUG)

NERSC Workload Diversity



6,000 users

850 projects

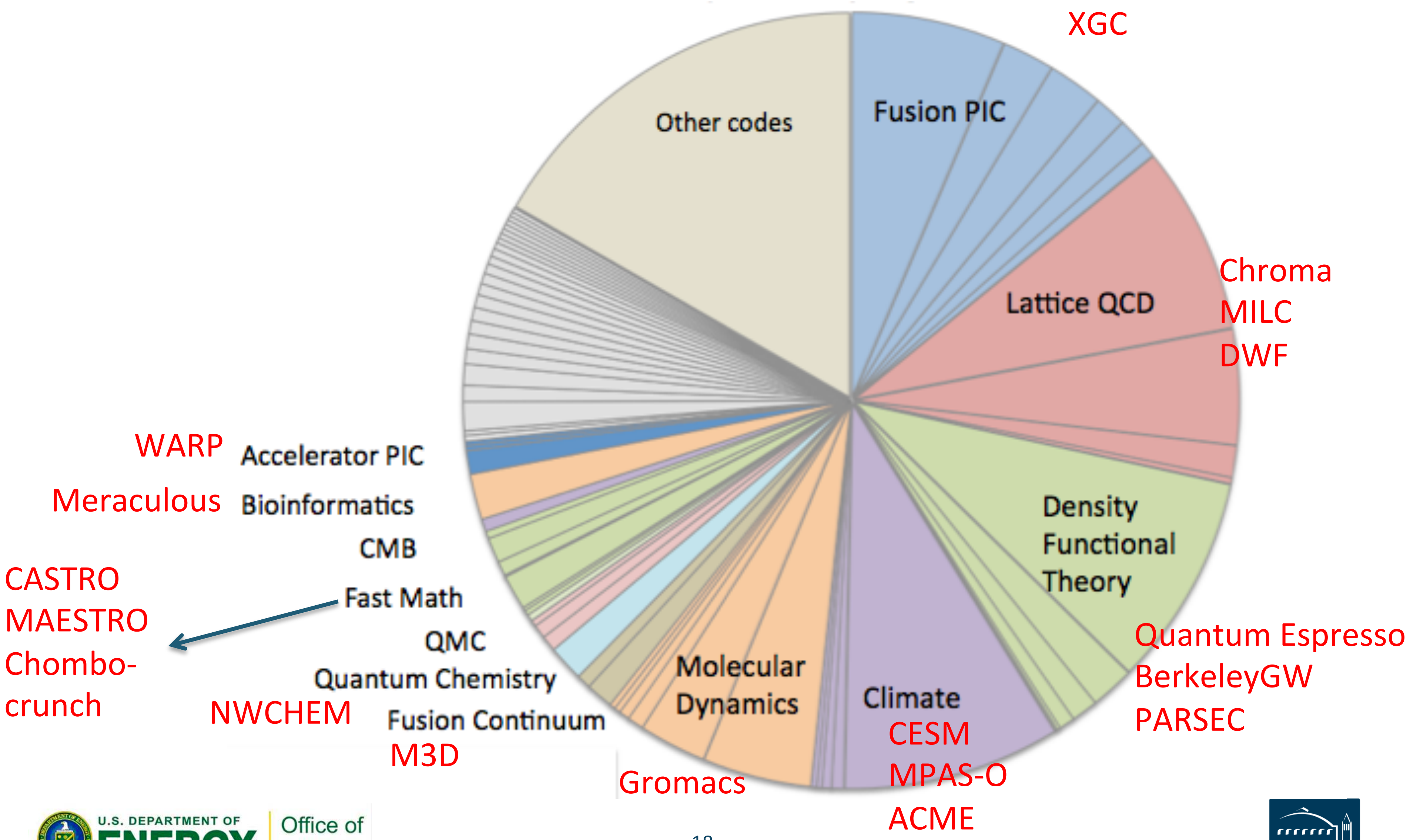
10 codes = 45% of cycles

25 codes = 66% of cycles

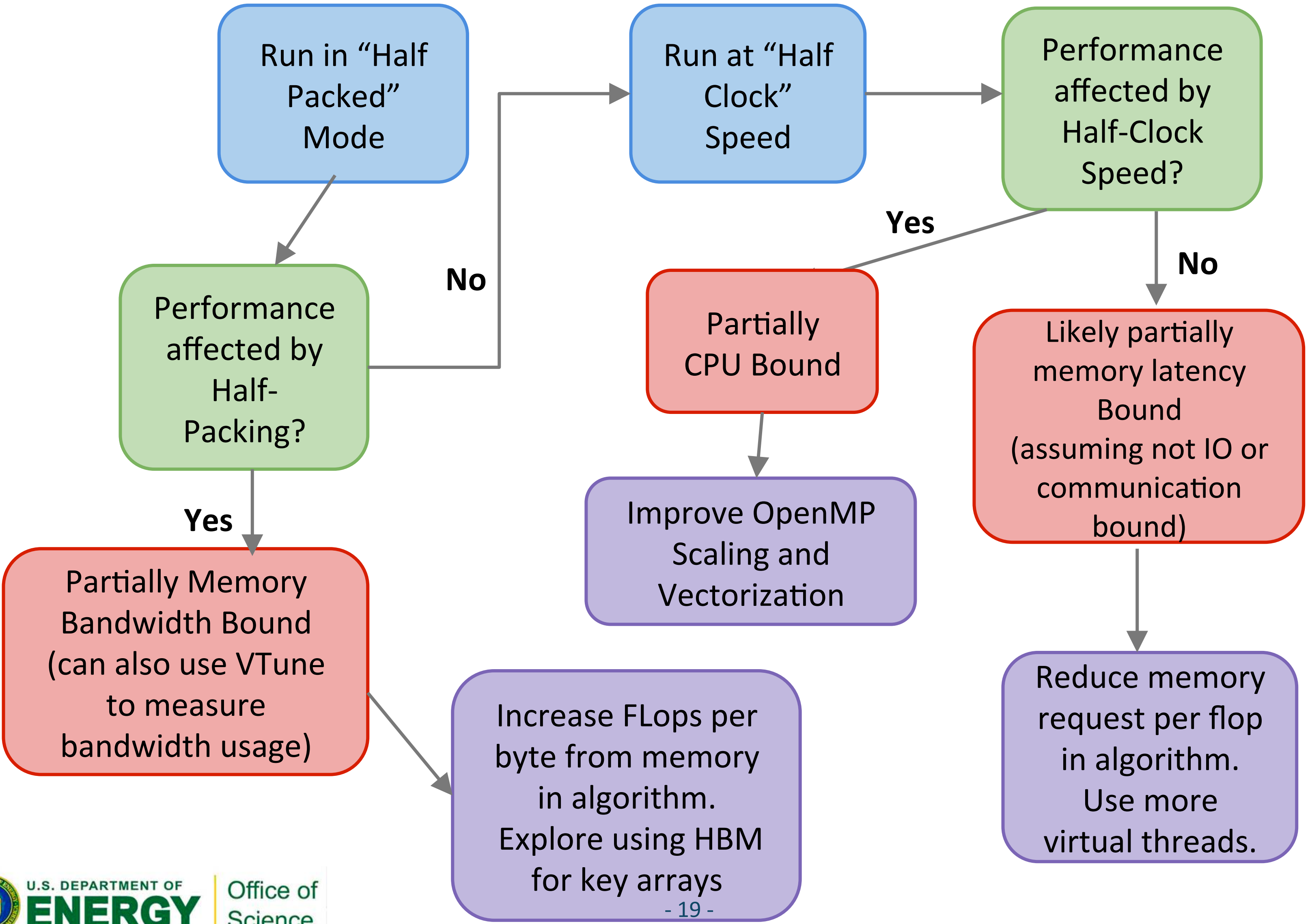
50 codes = 80% of cycles

"Other" > 600 codes (20%)

NESAP Applications represent wide range of algorithms



Recommended Optimization Path

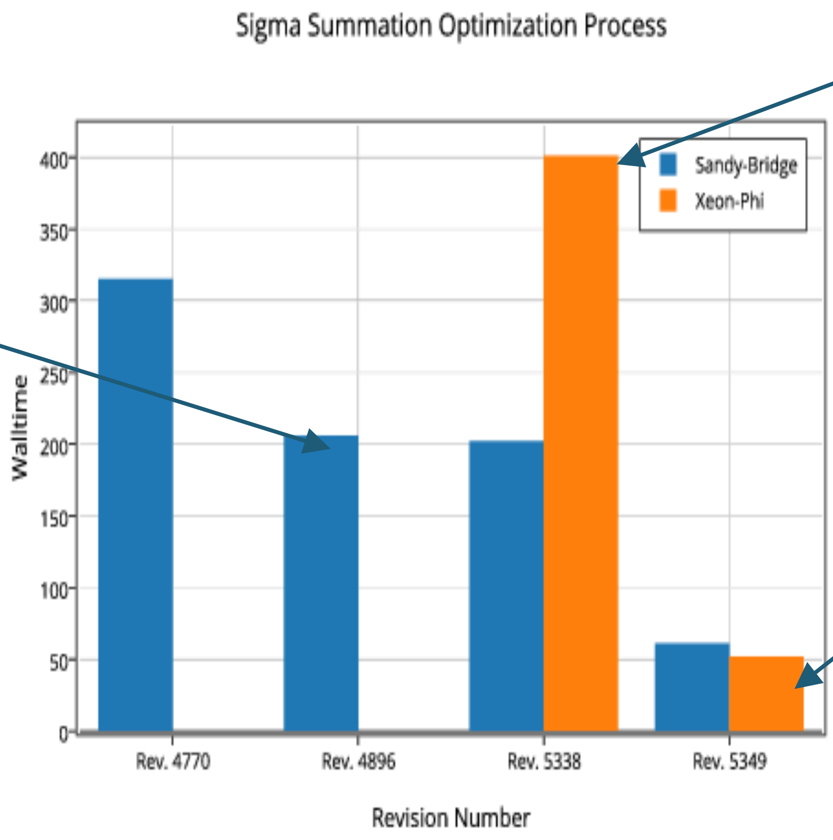

Run in "Half Packed" Mode
Performance affected by Half-Packing?
No
Run at "Half Clock" Speed
Performance affected by Half-Clock Speed?
Yes
Partially CPU Bound
Improve OpenMP Scaling and Vectorization
No
Likely partially memory latency Bound (assuming not IO or communication bound)
Reduce memory request per flop in algorithm. Use more virtual threads.
Yes
Partially Memory Bandwidth Bound (can also use VTune to measure bandwidth usage)
Increase FLops per byte from memory in algorithm. Explore using HBM for key arrays

BerkeleyGW Optimization Steps

- **Target more on-node parallelism. (MPI model already failing users)**
- **Ensure key loops/kernels can be vectorized.**

Refactor to Have 3 Loop Structure:

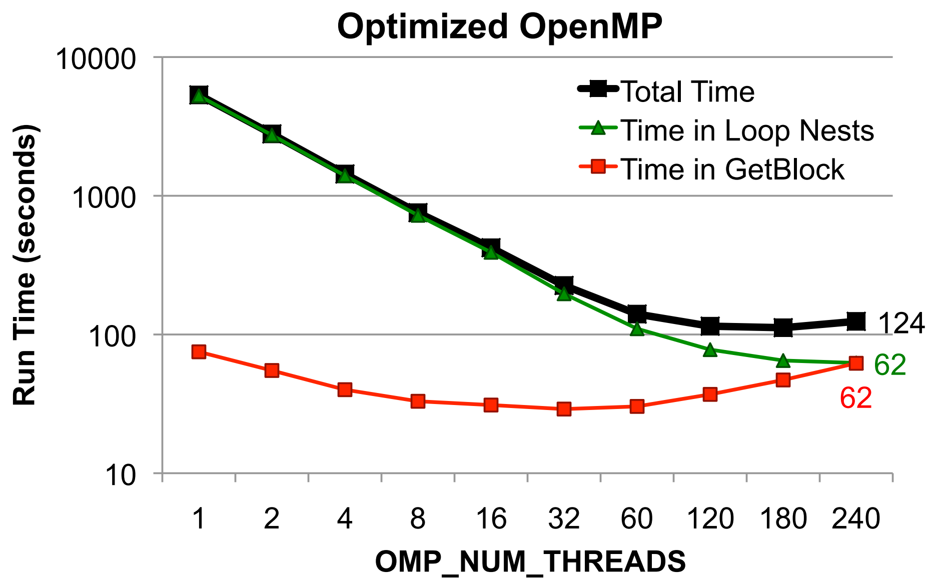
Outer: MPI
Middle: OpenMP
Inner: Vectorization



Add OpenMP

Ensure Vectorization

NWChem CCSD(T): OpenMP Optimizations



- GetBlock optimizations: parallelize sort, loop unrolling.
- Reorder array indices to match loop indices.
- Merge adjacent loop indices to increase number of iterations.
- Align arrays to 64 bytes boundary.
- Exploit OpenMP loop control directive, provide complier hints.
- Total speedup from base is 2.3x.

Courtesy of Hongzhang Shan et al., LBNL

Simulate HBM Effect on a Dual Socket System

- **Identify the candidate (key arrays) for HBM**
  - VTune Memory Access tool can help to find key arrays
  - Using NUMA affinity to simulate HBM on a dual socket system
  - Use FASTMEM directives and link with jemalloc/memkind libraries

On Edison (NERSC Cray XC30):

```
real, allocatable :: a(:,:), b(:,:), c(:)
!DIR$ ATTRIBUTE FASTMEM :: a, b, c
% module load memkind jemalloc
% ftn -dynamic -g -O3 -openmp mycode.f90
% export MEMKIND_HBW_NODES=0
% aprun -n 1 -cc numa_node numactl --membind=1 --cpunodebind=0 ./myexecutable
```

On Haswell:

```
% numactl --membind=1 --cpunodebind=0 ./myexecutable
```

Application	All memory on far memory	All memory on near memory	Key arrays on near memory
BerkeleyGW	baseline	52% faster	52.4% faster
EmGeo	baseline	40% faster	32% faster
XGC1	baseline		24% faster

Conclusions

- NERSC is bringing a lot of resources to help users: training, postdocs, Cray and Intel staff, deep dive sessions.
- Optimizing code for Cori KNL will likely require good OpenMP scaling, Vectorization and/or effective use of HBM.
- Applications can optimize on IvyBridge (Edison), Haswell (Cori P1), and KNC (Babbage) architectures to prepare for Cori.
- Always profiling and understand your code first on where to work on improving performance. Use tools such as VTune and vector advisor.
- Creating kernels is much more efficient than working on full codes.
- Optimizing your code targeting KNL will improve performance on all architectures.
- Keep portability in mind, use portable programming models.

NERSC 40 YEARS at the FOREFRONT

Thank you.

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB
Lawrence Berkeley National Laboratory