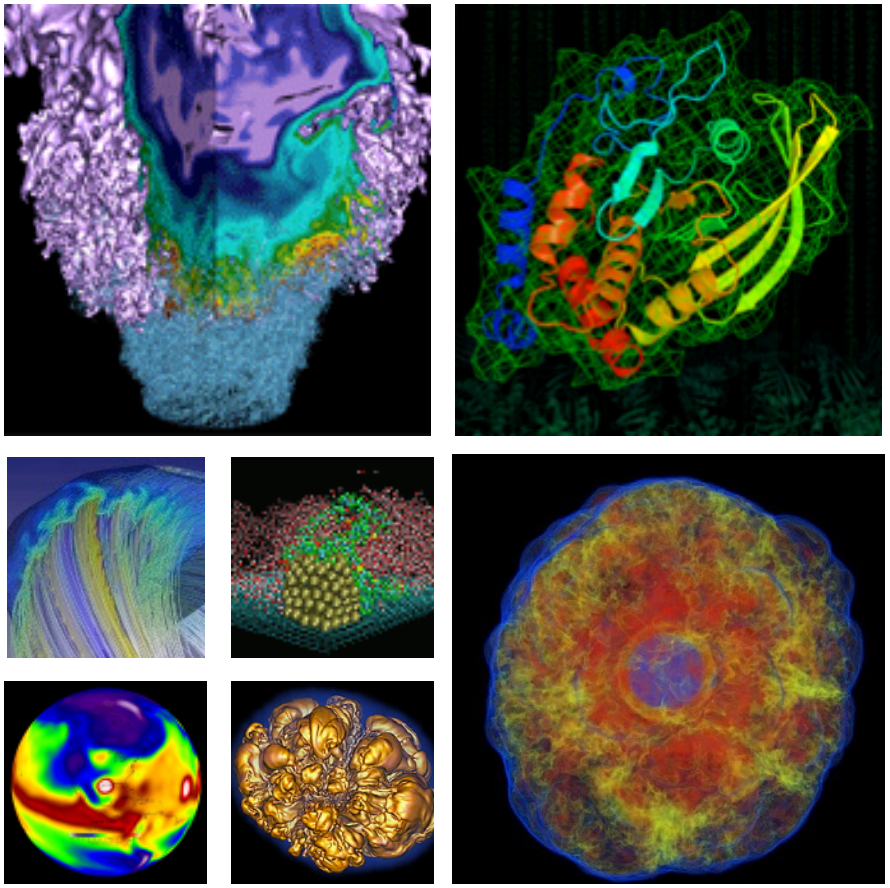
# Analyzing Performance of Selected NESAP Applications on the Cori HPC System



T. Kurth, W. Arndt, T. Barnes, B. Cook, J. Deslippe, D. Doerfler, B. Friesen, Y. He, T. Koskela, M. Lobet, T. Malas, L. Oliker, A. Ovsyannikov, S. Williams, W-S. Yang, Z. Zhano

ISC-IXPUG 2017 Frankfurt, Germany
June 21, 2017

NERSC

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB

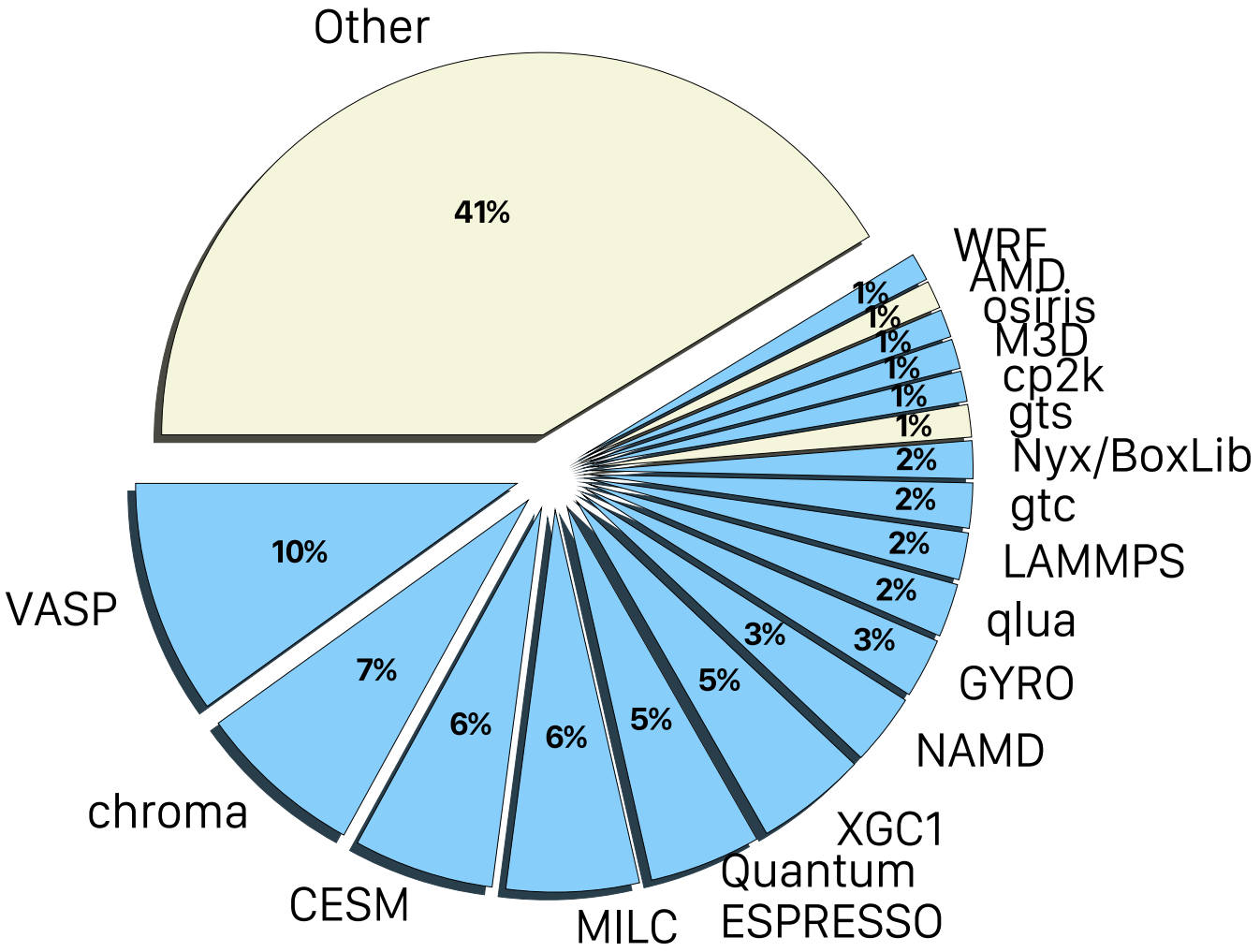
# NERSC Overview

- NERSC statistics:
  - production HPC facility for DOE
  - more than 6000 users worldwide
  - diverse user-base and science domains: astrophysics, chemistry, materials science, nuclear physics, high-energy physics, climate, geosciences, biology, fusion
  - challenge: support a huge variety of codes
- upcoming: data-intensive applications enter the mix

# NESAP

- NERSC transition: conventional multi-core to energy-efficient many-core Intel Xeon Phi 7250 (Knights Landing)
- facilitate this transition through Nersc Exascale Science Application Program
- main NESAP goals:
  - hands-on improvements of user-codes in coordination with domain scientists
  - educating users (lessons learned) with hands-on training, presentations and case studies

# NERSC 2015 Code Usage



# Code Coverage

| App | Description | Proxy | Kernels |
|---|---|---|---|
| BerkeleyGW | MBPT | - | FFT, LA |
| GROMACS | Molecular Dynamics | LAMMPS, NAMD | Force Calculation |
| Qbox | PW DFT | cp2k | FFT, LA, Eigensolver(lanczos) |
| Quantum ESPRESSO | PW DFT | - | FFT, LA Eigensolver(lanczos) |
| VASP | PW DFT | - | FFT, Eigensolver(multiple) |
| WARP | PIC | osiris | Gather, Sort, FFT, Solver |
| XGC1 | PIC | gtc, gts, GYRO | Gather, Sort |
| Chombo-Crunch | AMR EB | - | EB Stencil(3D), Solver(AMG) |
| Nyx/BoxLib | AMR | - | Stencil(3D), Solver(GMG) |

# Code Coverage

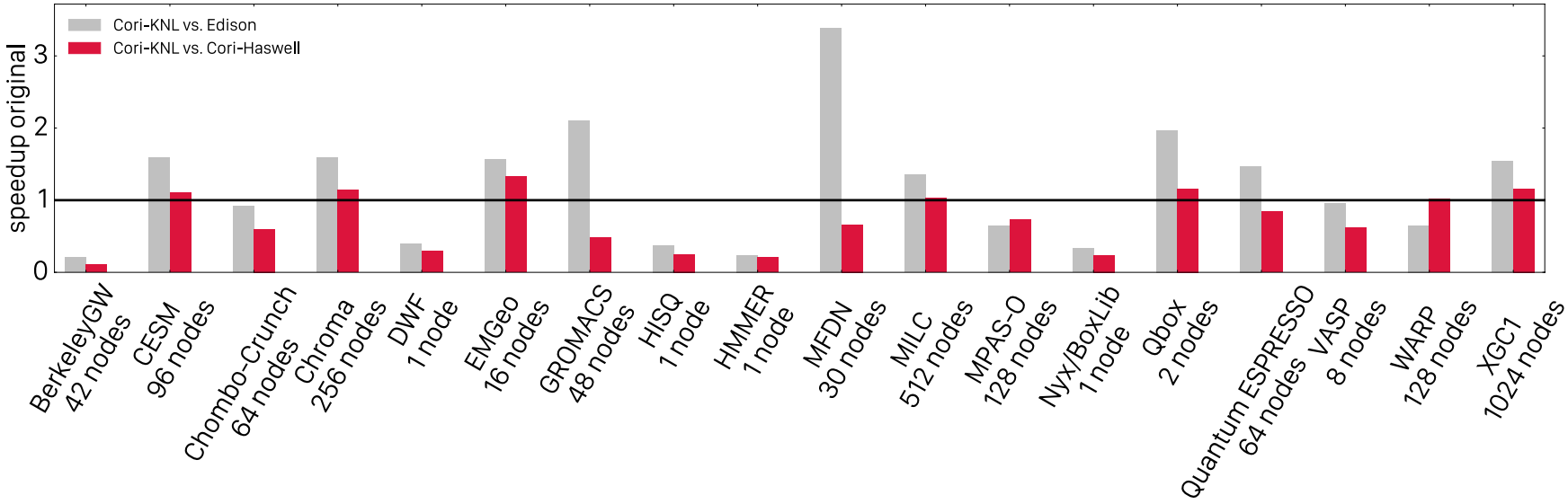
| App | Description | Proxy | Kernels |
|---|---|---|---|
| **CESM** | Grid | WRF | Stencil(multiple), LA |
| **MPAS-O** | Unstructured Grid | WRF | Gather, Solver(RK4) |
| **Chroma** | Lattice QCD | qlua | Stencil(4D), Solver(BiCGStab) |
| **DWF** | Lattice QCD | qlua | FFT, Stencil(5D), Solver(BiCGStab) |
| **HISQ** | Lattice QCD | - | Stencil(4D), Solver(BiCGStab) |
| **MILC** | Lattice QCD | qlua | Stencil(4D), Solver(BiCGStab) |
| **EmGeo** | Grid | - | SpMM, Solver(IDR) |
| **HMMER** | Gene Annotation | - | Dynamic Programming(2D), Byteword Arithmetics |
| **MFDN** | Many Body | - | SpMM, Eigensolver(lanczos) |

# Xeon Phi 7250 Features

- important hardware features
  - 68 cores@1.2 Ghz, 272 threads in total
  - complicated memory topology: configurable 2D on-chip interconnect, shared L2 cache per tile, no L3
  - 512bit-wide vector units with FMA support and additional fast reduced precision intrinsics
  - 16 GB high-bandwidth on-package memory (HBM/ MCDRAM), configurable as cache, flat or hybrid cache/flat
- KNL-ready applications should exploit (some of) these features

# Recompile and Run

- x86-64 compatible
- self-hosted

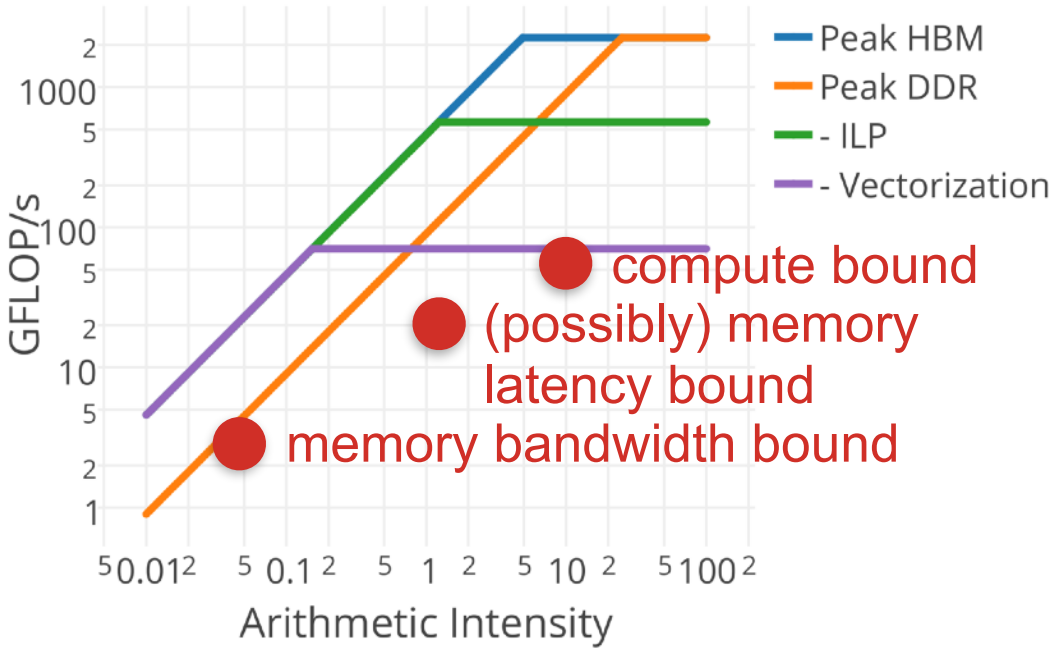



- median speedup vs. Ivy-Bridge 1.2x, vs. Haswell 0.7x
- most codes need (some degree of) optimization

# Optimization Process - Roofline Model

- employ roofline model to guide optimization efforts
  - identify hotspots
  - measure arithmetic intensity AI
  - display on roofline
  - decide next steps
- helps deciding „when to stop“
- but: most efforts ongoing

AI: #FLOPS/#BYTES
P: #FLOPS/time



# Most Promising Optimizations

- **identifying and exploiting parallelism**: create more work for individual threads (reduce OpenMP overhead), use `omp collapse`, transition to *coarse grained parallelization*
- **loop tiling**: missing L3 can hurt severely, *block loops to shared L2* (512KiB/core), also improves performance on multicore
- **short loop unrolling**: helps compiler to vectorize the right loops
- **ensuring efficient vectorization**: loop reordering/restructuring, data layout changes, can help latency bound codes because of more efficient prefetching, *read optimization reports carefully*
- **optimized mathematical functions**: make use of new ISA features (`-no-prec-div -fp-model fast=2`)

# Network Related Optimizations

- **hugepages** can reduce Aries TLB misses
- MPI-collective-heavy codes: **enable DMAPP** (add `-ldmapp`)

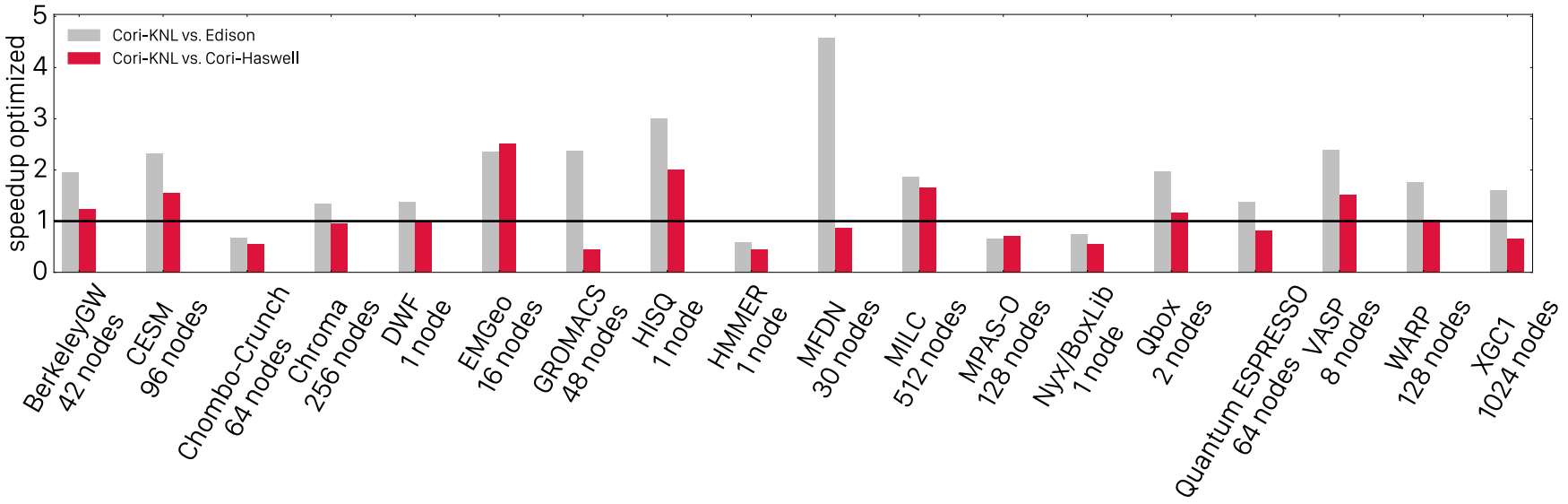
```
export MPICH_RMA_OVER_DMAPP=1
export MPICH_USE_DMAPP_COLL=1
export MPICH_NETWORK_BUFFER_COLL_OPT=1
```

- **non-blocking communications**
- enable hardware AMO for MPI-3 RMA atomics

```
export MPICH_RMA_USE_NETWORK_AMO=1
```

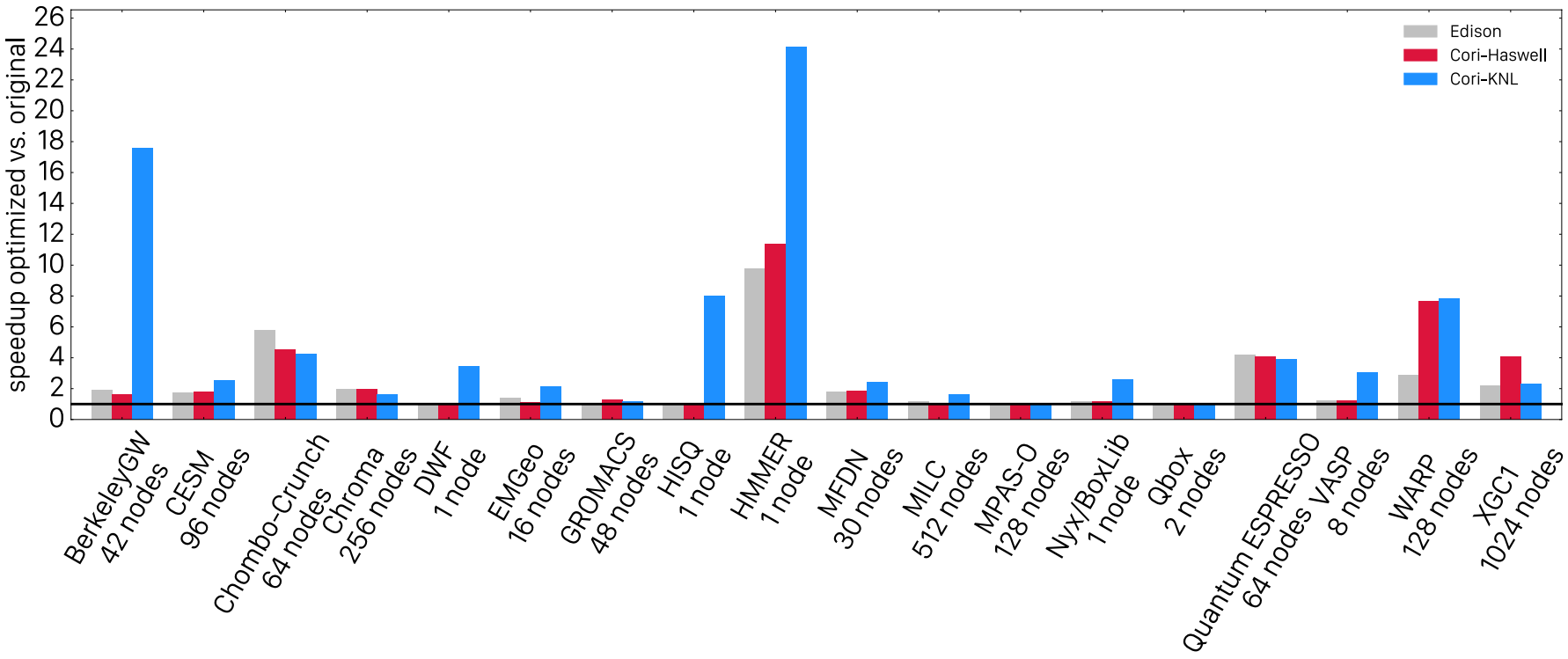
- not KNL-specific, all NERSC systems have Aries interconnect

# Speedup Optimized KNL vs. Multicore



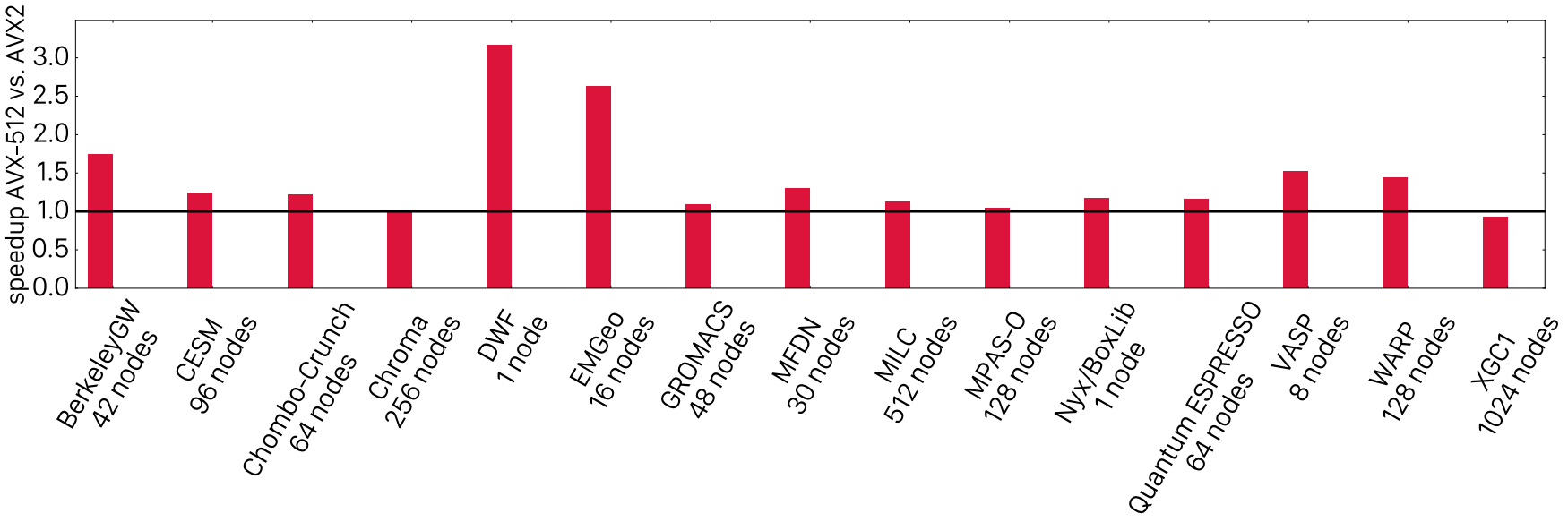

- median speedup vs. Ivy-Bridge 1.8x, vs. Haswell 1.0x
  - *worst codes*: memory latency sensitive, hard to parallelize
    - irregular memory access, pointer chasing (Chombo-Crunch),
    - hard-to-vectorize adaptive algorithms (stiff ODE integrator, Nyx)
    - load-balancing (HMMER)

# Speedup per Architecture



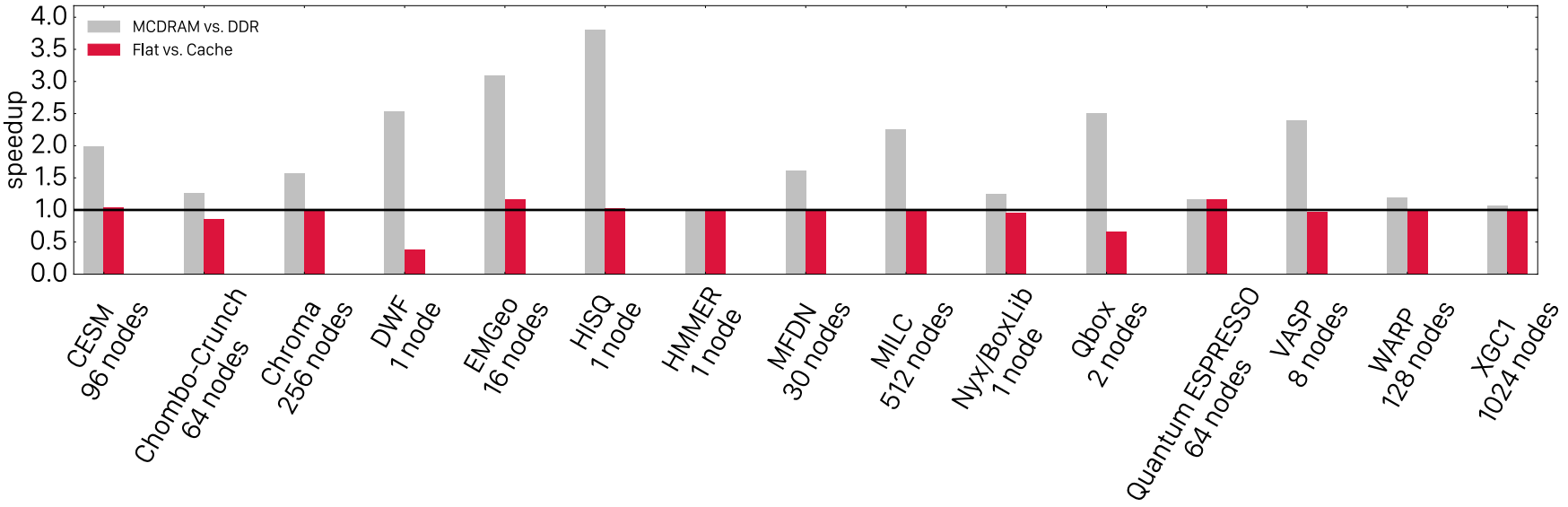

- speedups on all architectures, most significant on KNL
- Ivy-Bridge: 1.6x, Haswell: 1.5x, KNL: 2.6x

# AVX512 vs. AVX-2



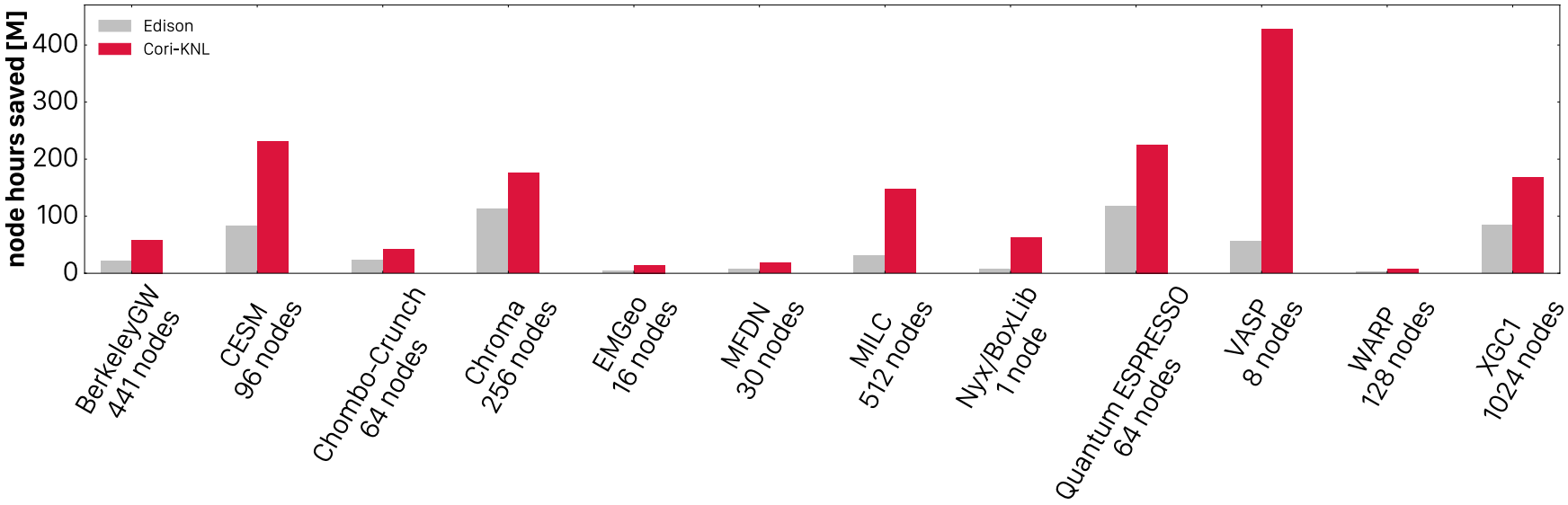

- median speedup: 1.2
- speedups >2: more efficient prefetching due to new ISA
- always use AVX512 instructions

# Memory Performance Comparison




- median speedup: MCDRAM 1.6x, flat 1.0x
- cache mode performs well for many applications, even multi-node
- for flat mode: either fit in or manage manually with directives/AutoHBW (relying on `numactl -p 1` is generally not a good idea)

# Savings per Machine



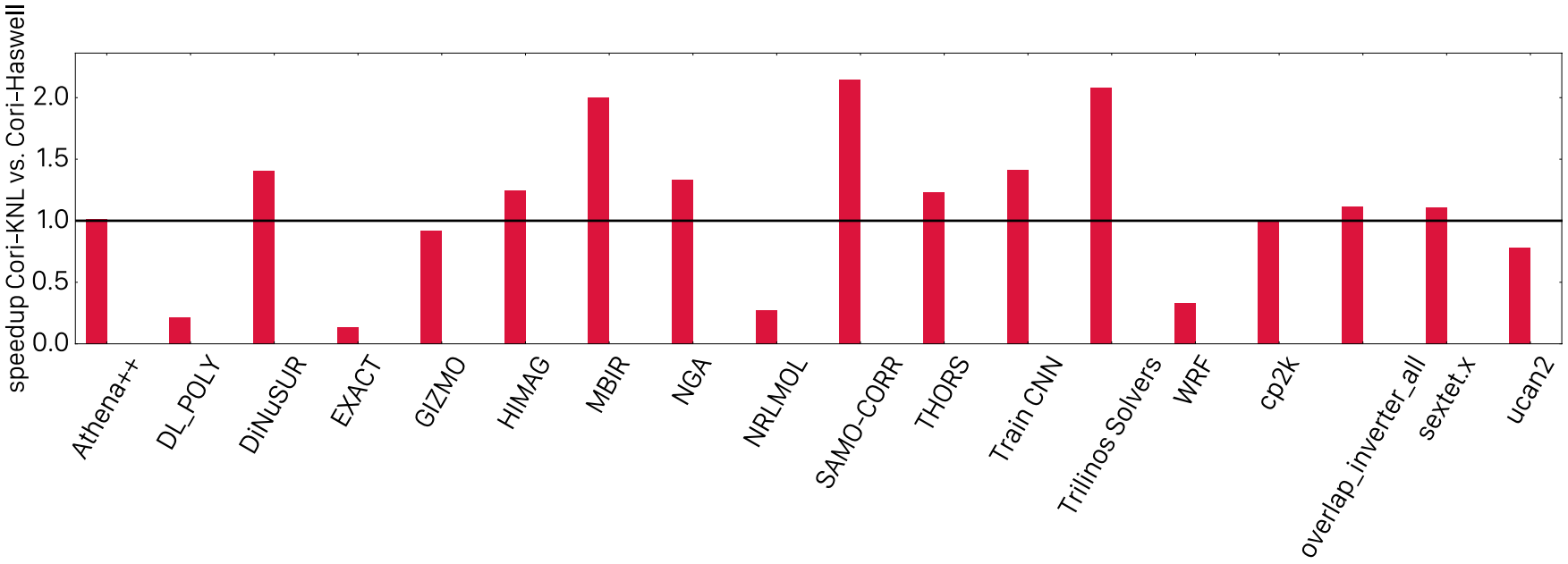

- total savings: Edison 550Mh, Cori-KNL 1577Mh
- assumptions:
  - use cases similar to optimized ones
  - Edison usage fraction carries over to Cori-KNL
  - proxy-apps not included

# Summary

- most promising optimizations
  - exploit parallelism
  - loop fusion, tiling and unrolling
  - ensure good vectorization
  - use MCDRAM
  - non-blocking communications and hugepages
- different codes benefit differently from these techniques
- some techniques enable others:
  - improved data layout can enable vectorization
  - good vectorization helps mitigating memory latency

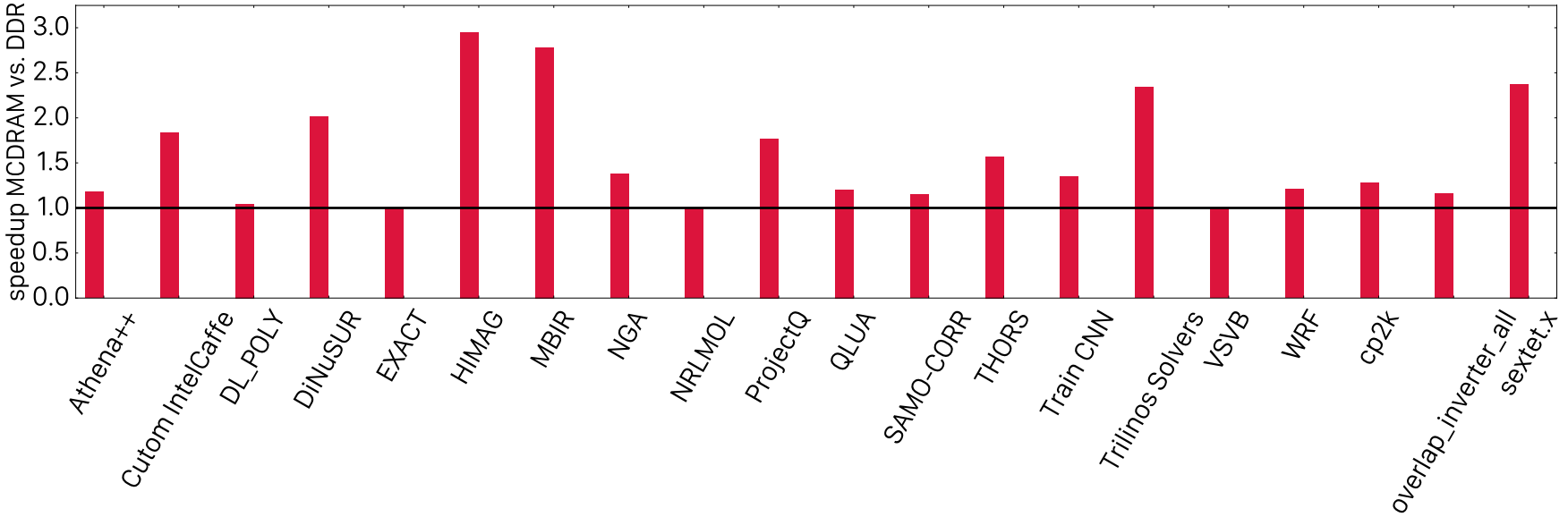

# Thank you

# Speedup KNL vs. Multicore non-NESAP




- median speedup vs. Haswell 1.1x

# Memory Performance Comparison non-NESAP




- median speedup: MCDRAM 1.6x