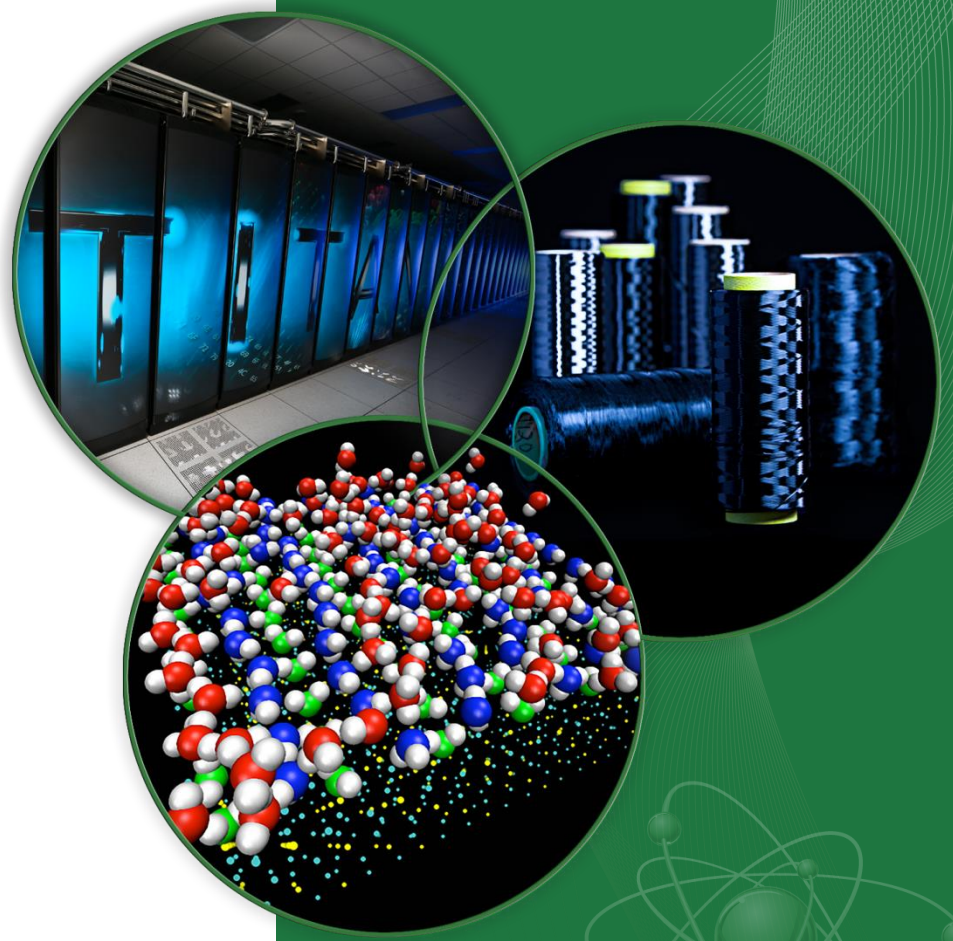
Examining Recent Many-core Architectures and Programming Models Using SHOC

M. Graham Lopez
Jeffrey Young
Jeremy S. Meredith
Philip C. Roth
Mitchel Horton
Jeffrey S. Vetter

PMBS15

Sunday, 15 Nov 2015



ORNL is managed by UT-Battelle
for the US Department of Energy

OAK RIDGE National Laboratory

Answering Questions about Heterogeneous Systems

- How does one device perform relative to another?
- In which areas is one accelerator better?
- How do multiple devices perform (separately or in concert)?
- How do heterogeneous programming models compare?
- What's the most productive way to program a given device?

SHOC 1.0

OAK RIDGE
National Laboratory

Scalable Heterogeneous Computing Suite

- Benchmark suite with a focus on scientific computing workloads
- Both performance and stability testing
- Supports clusters and individual hosts
  - intra-node parallelism for multiple GPUs per node
  - inter-node parallelism with MPI
- Both CUDA and OpenCL
- Three levels of benchmarks:
  - Level 0: very low-level device characteristics (bus speed, max flops)
  - Level 1: low level algorithmic operations (fft, gemm, sorting, n-body)
  - Level 2: application-level kernels (combustion chemistry, clustering)

A. Danalis, G. Marin, C. McCurdy, J.S. Meredith, P.C. Roth, K. Spafford, V. Tipparaju, J.S. Vetter
"The Scalable Heterogeneous Computing (SHOC) Benchmark Suite"
Third Workshop on General-Purpose Computation on Graphics Processing Units (GPGPU-3), 2010.

https://github.com/vetter/shoc

SHOC 2.0

OAK RIDGE National Laboratory

Recent Additions to SHOC

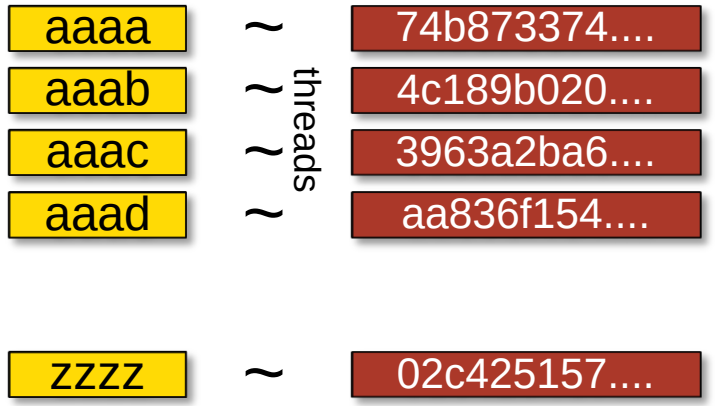
- Added new benchmarks
  - Originals focused on floating point, scientific computing applications
  - New benchmarks: machine learning, data analytics, and integer operations
- Supports new programming models
  - Original supported OpenCL when it was new
    - Allowed CUDA vs OpenCL comparisons
    - Multiple OpenCL implementations could support one platform
    - Tracking maturity of OpenCL over time
  - New programming models support directives
    - OpenACC, OpenMP + offload
    - Better support for multi-core and new devices (Intel Xeon Phi)

New Benchmarks

OAK RIDGE National Laboratory

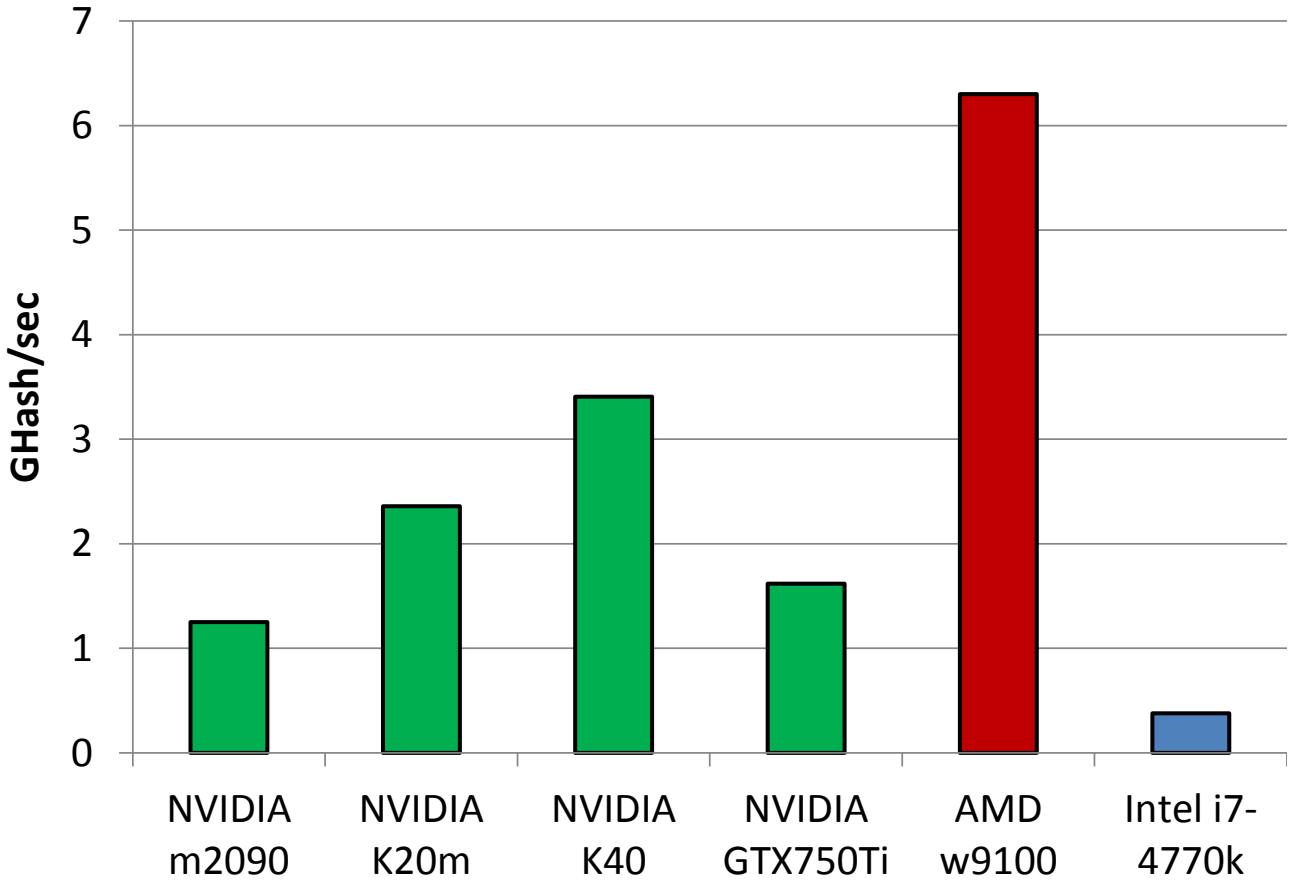
MD5Hash

- MD5 is a cryptographic hash function
  - Heavy use of integer and bitwise operations
  - No floating point operations
- Not parallel for a single input string
  - Would be bandwidth-dependent to be useful anyway
- Instead, do a parallel search for a known, random hash
  - Each thread hashes a large set of short input strings
  - Input strings are generated programmatically from a given key space



MD5Hash Results

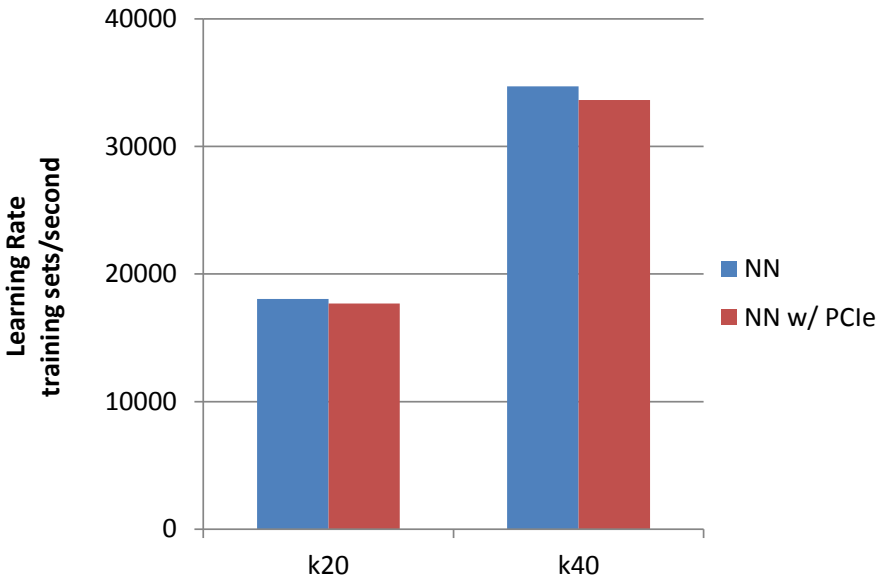
- Large generational improvements for NVIDIA
  - Kepler K40 vs Fermi m2090 almost 3x
  - Maxwell 750Ti outperforms Fermi m2090
- AMD better overall for integer/bit operations
  - w9100 vs k40 almost 2x



Neural Net (NN)

- Neural Net is represented by a deep learning algorithm that can identify pictures of handwritten numbers 0-9 from MNIST inputs
  - CUDA version with CUBLAS support
  - Phi/MIC version with OpenMP/offload support
    - Limited MKL use; rectangular matrices impact threading
- 784 input neurons, ten output neurons, and one hidden layer with thirty neurons
  - 50,000 training sets

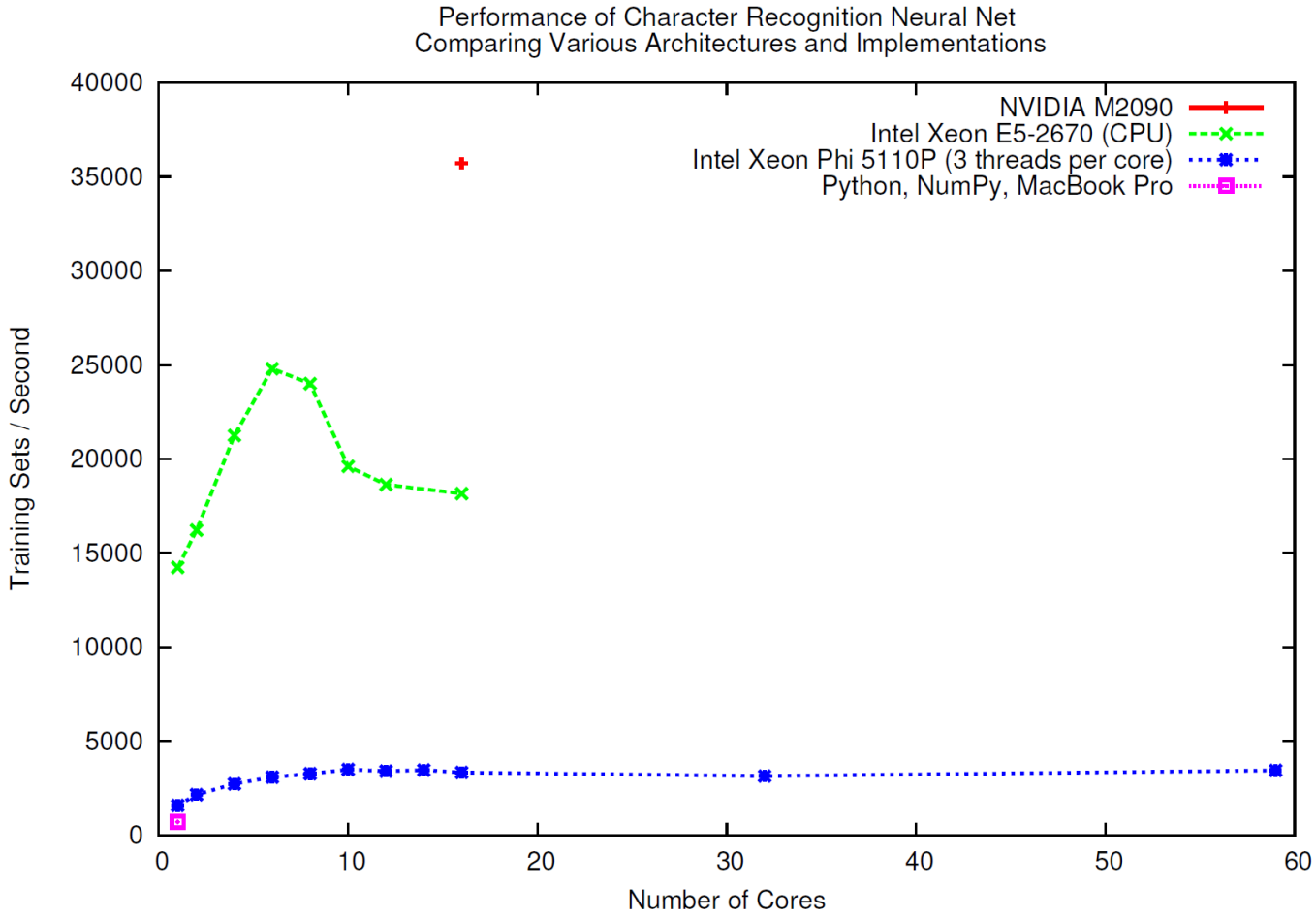
Visualization of Testing Set [3]



[1] M. Nielsen. Neural networks and deep learning. October 2014. https://github.com/mnielsen/neural-networks-and-deep-learning.
[2] Y. LeCun, C. Cortes, and C. J. Burges. The MNIST database of handwritten digits. 2014. http://yann.lecun.com/exdb/mnist/.
[3] http://eblearn.sourceforge.net/mnist.html

Neural Net Results

- CUBLAS is well tuned for rectangular matrices
  - m2090 outperforms all others
- MKL does not use threads for these matrices
  - Custom OpenMP code
  - ... but was not well vectorized by the compiler
  - Poor thread scaling on Xeon Phi limits its performance

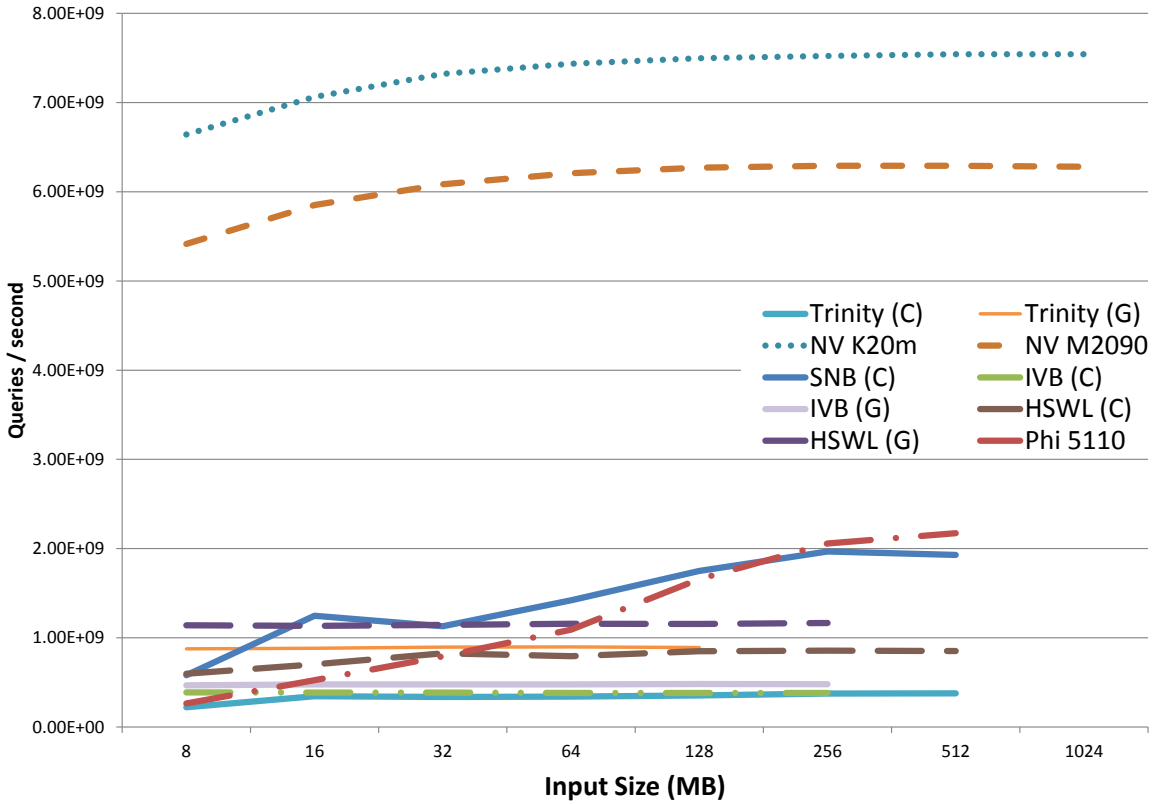


Data Analytics

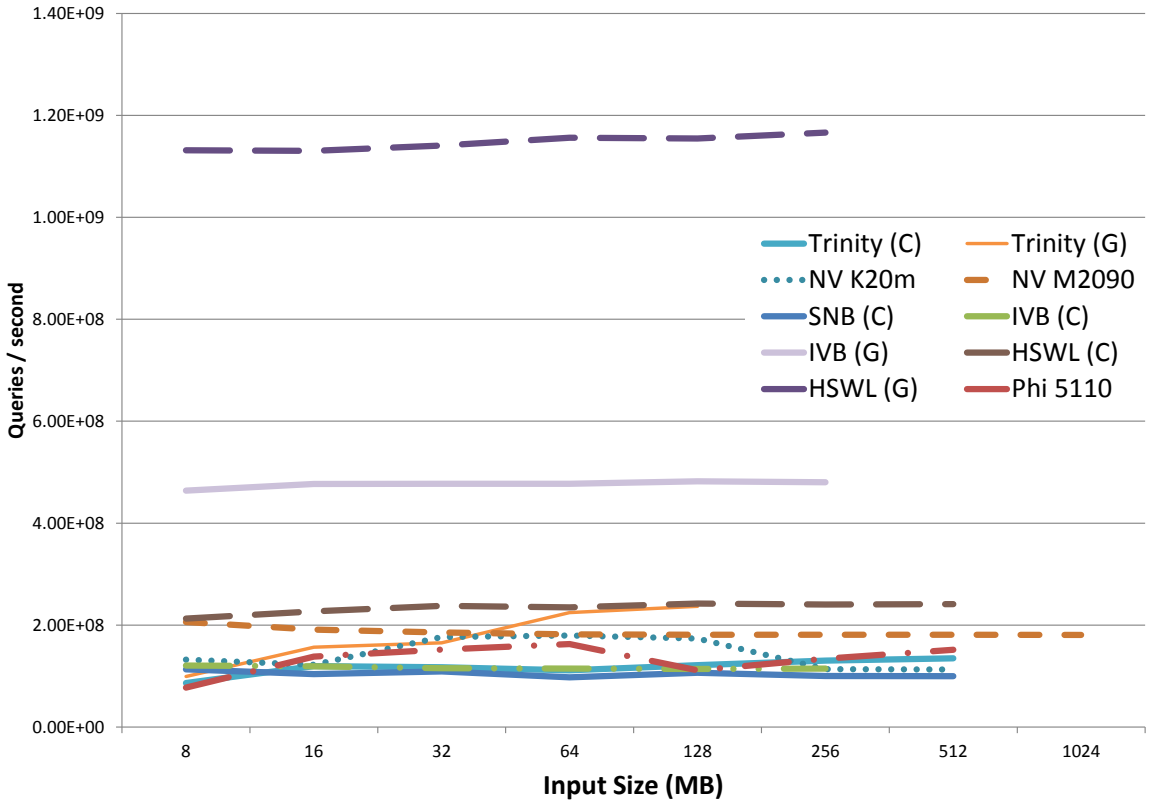
- Data analytics is represented by relational algebra kernels like Select, Project, Join, Union
  - These kernels form the basis of read-only analytics for benchmarks like TPC-H [1] that have been accelerated with CUDA [2].
- SHOC's OpenCL implementation allows for testing on CPU, GPU, and Phi without needing a large database input
  - All tests are standalone with randomly generated tuples
  - More information on the implementation in related work [3]

[1] T. P. P. Council. TPC Benchmark H (Decision Support) Standard Specification, Revision 2.17.0. 2013. http://www.tpc.org/tpch/
[2] H. Wu, G. Diamos, S. Cadambi, and S. Yalamanchili. *Kernel weaver: Automatically fusing database primitives for efficient GPU computation*. *MICRO 2012*
[3] Ifrah Saeed, Jeffrey Young, Sudhakar Yalamanchili, *A portable benchmark suite for highly parallel data intensive query processing. PPAA 2015*

Data Analytics Results



Project, no PCIe Transfer Time



Project, Transfer Time Included

- Kepler GPU performs best with 7.54 giga-ops/second (GOPS); sensitivity to tuning parameters (like workgroup size) makes performance portability difficult for this code
- Haswell GPU has the best performance when data transfer is included – 1.17 GOPS for 256 MB input; Haswell GPU has the best "zero-copy" semantics of integrated GPUs

New Programming Models

Programming Models

- Originally: CUDA, OpenCL
- Added: OpenACC, Xeon Phi (OpenMP and LEO)
- Planned: pure OpenMP
  - When compilers support accelerator features
- Examples often compare directives to lower-level
  - Directives aren't expected to outperform, but how much of a loss?
  - What are the other issues (if any)?

SHOC Example Studies

SHOC Example Studies

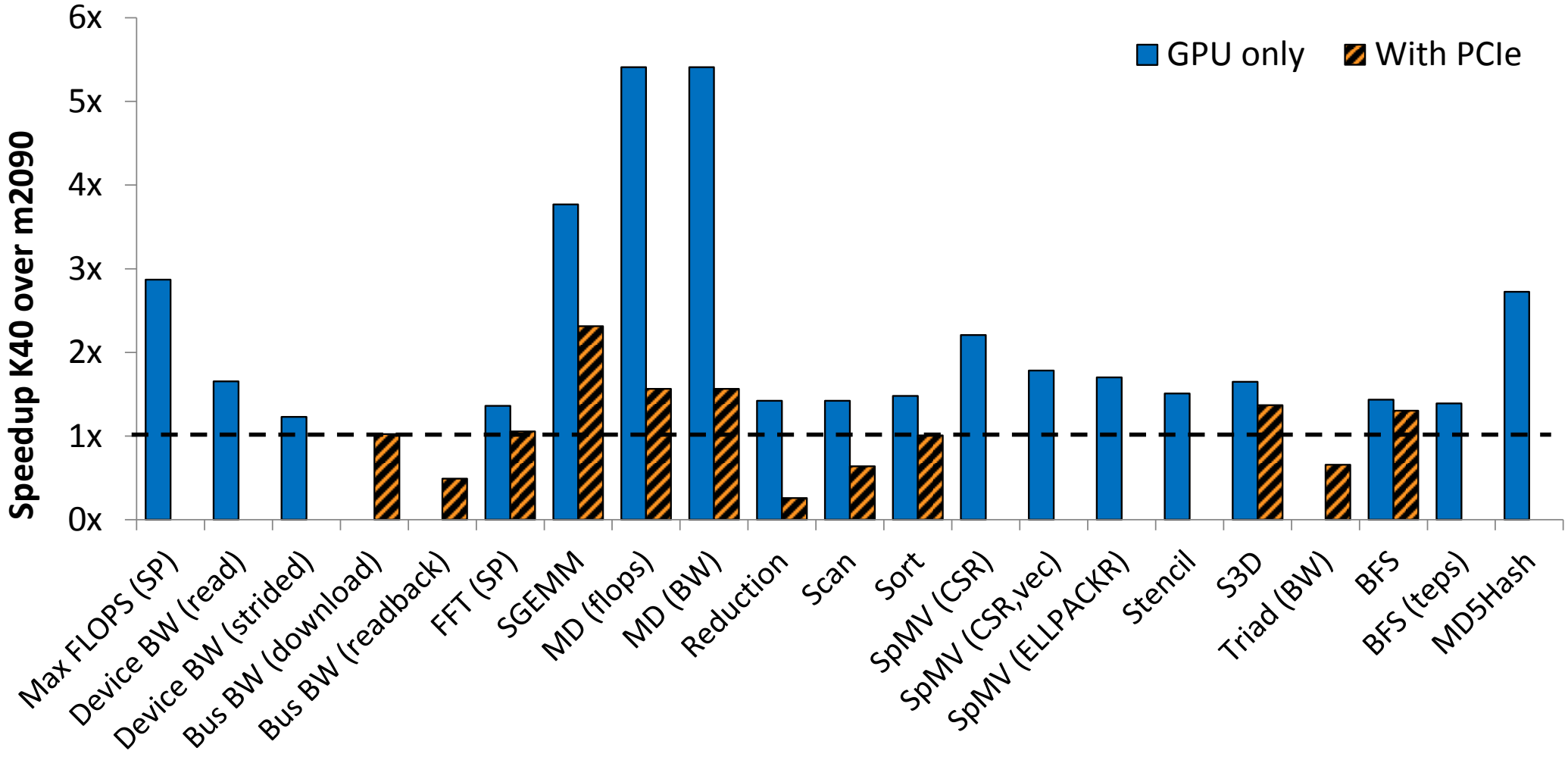
- SHOC can be useful for understanding:
  - heterogeneous and many-core system hardware
  - programming heterogeneous systems and accelerators
- To explore the space of potential studies, we show:
  - Example hardware comparisons
  - Example programming model comparisons
- These are example analyses to show possibilities
  - Breadth more than depth
  - Others may ask and answer entirely new questions using SHOC

Hardware Comparisons

SHOC Example Hardware Studies

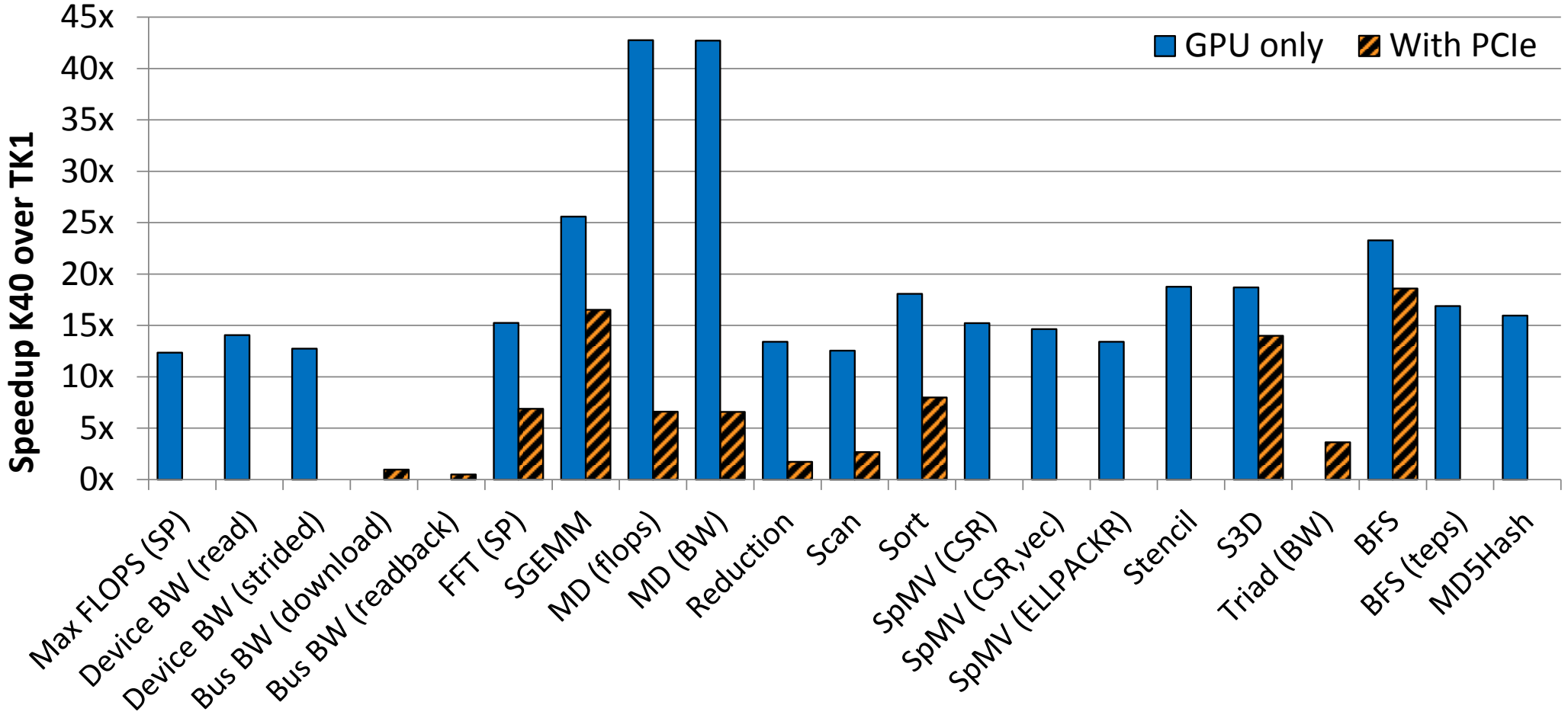
- Generational improvements for same vendor
  - NVIDIA Fermi m2090 vs Kepler K40
- Large vs small device in same architectural line
  - NVIDIA K40 (15 SMX) vs Jetson TK1 (1 SMX)
- Cross-vendor, i.e., different architectures
  - NVIDIA K40 vs AMD w9100
  - NVIDIA K20 vs Intel Xeon Phi (KNC)

Generational Improvement for Same Vendor



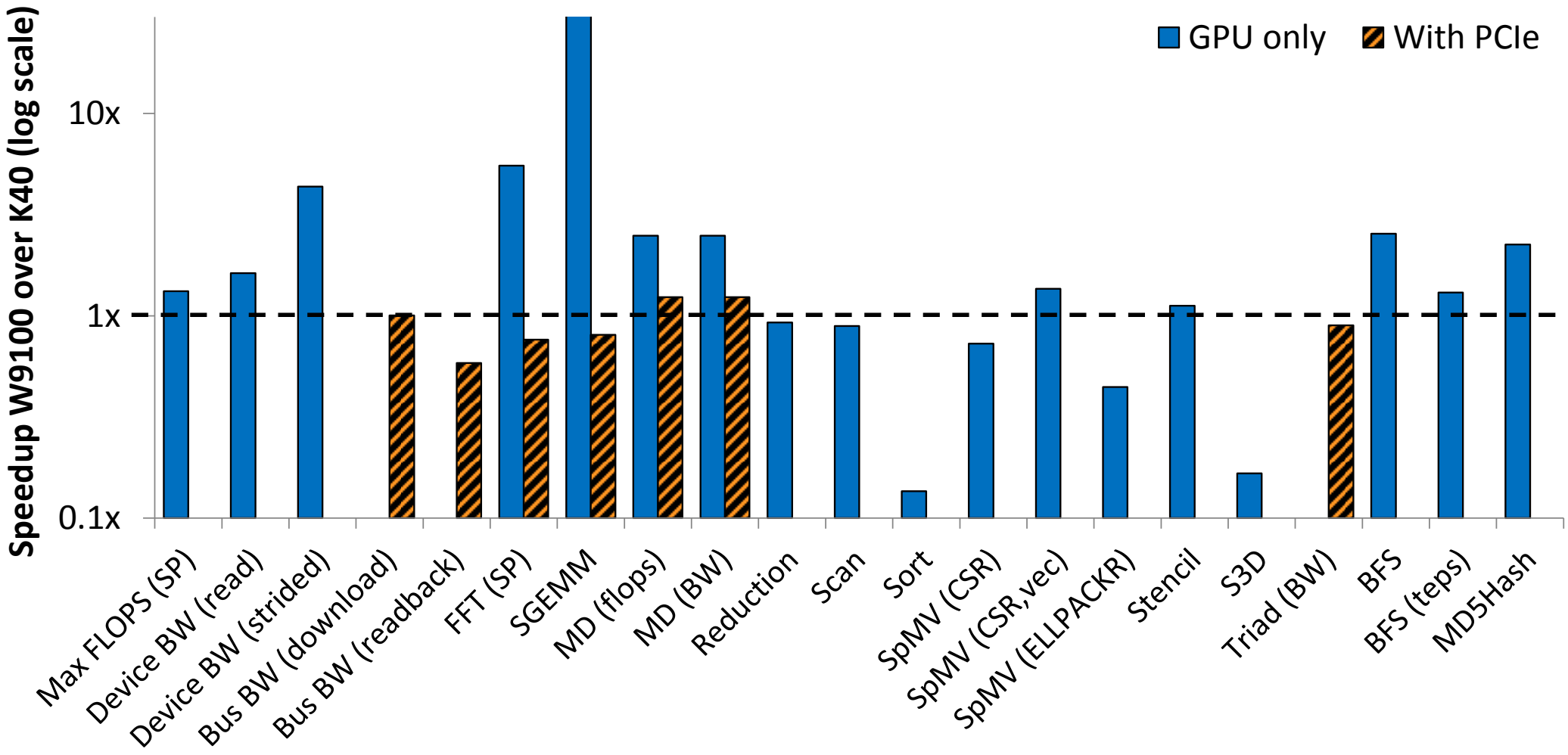
- Host platform differences limited bus speed and impacted PCIe results on newer device

Large vs Small Device of Same Architecture



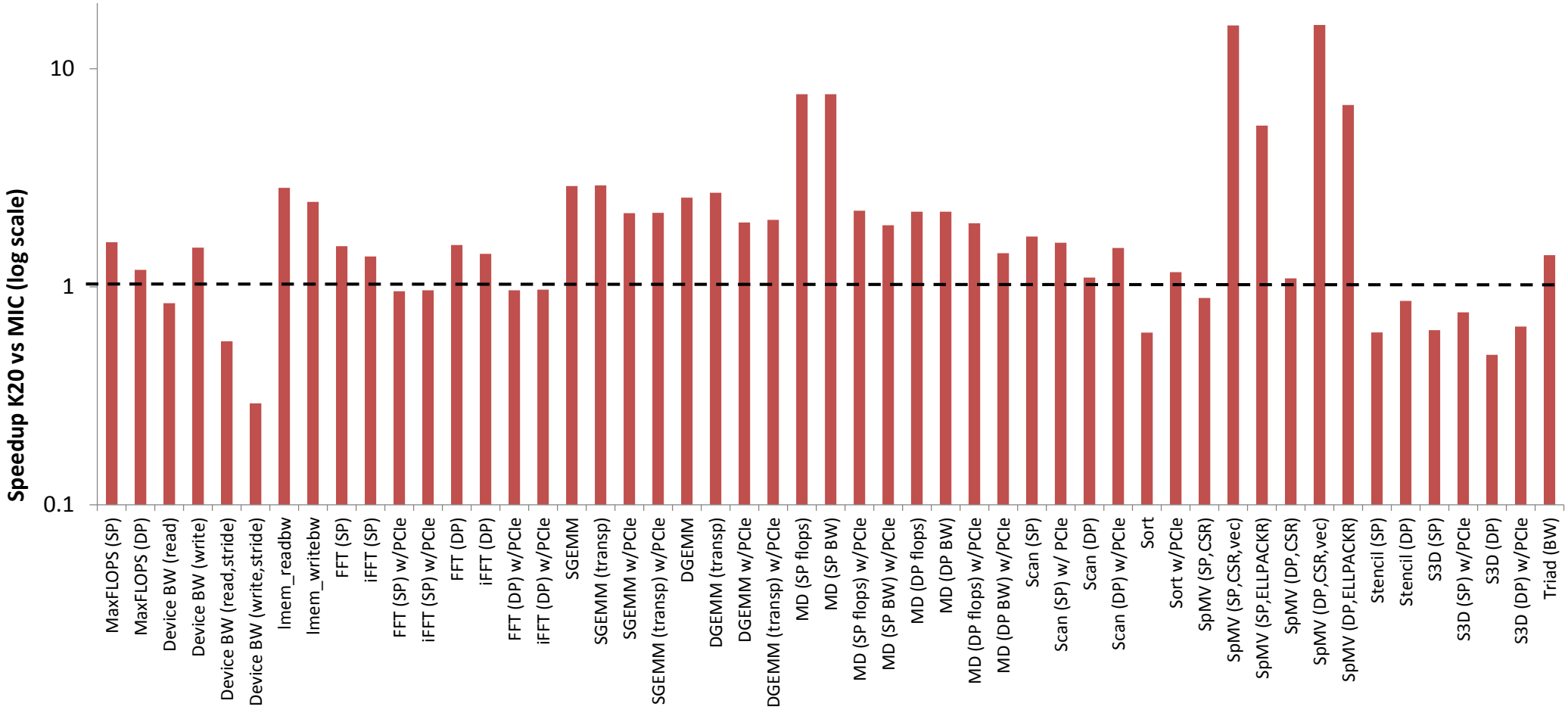
- 15:1 raw SMX ratio. Accounting for clockspeeds, expect core=14:1, bandwidth=12:1
- Similar host-device speed limits improvement in “PCIe” benchmarks
- Unexpected K40 improvements (host/platform, library optimization, or other HW differences)

Cross-Vendor Comparisons (AMD v NVIDIA, OpenCL)



- Raw (level 0) numbers generally better for W9100, translated into several AMD wins
- Integer performance on W9100 relatively better (MD5Hash) versus floating point

Cross-Vendor Comparisons (NVIDIA v Intel)



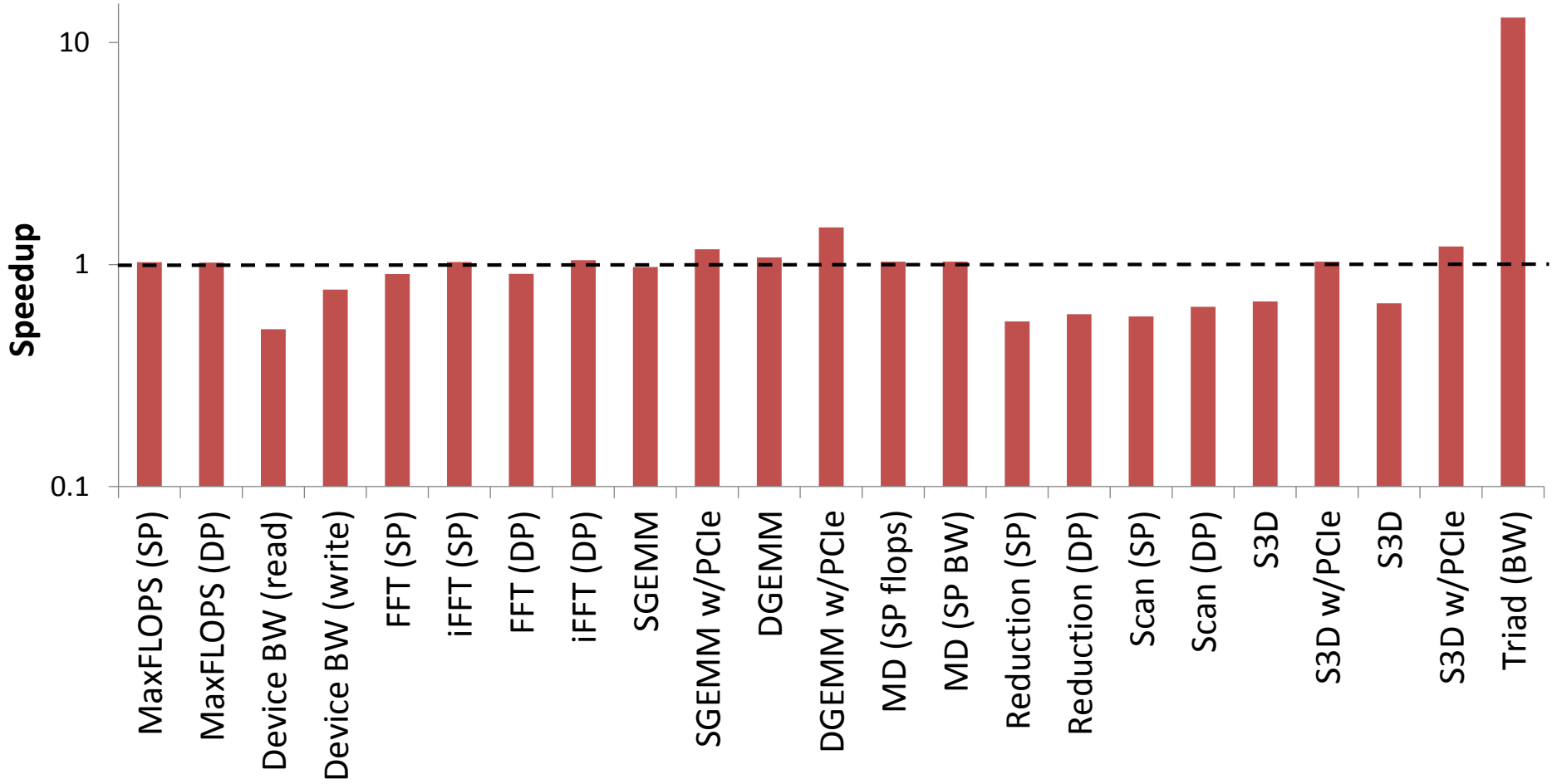
- Xeon Phi double precision is relatively better than K20 (i.e. bigger win/smaller loss in DP vs SP)
- Cache size vs local memory effects have complex tradeoffs

Programming Model Comparisons

SHOC Example Programming Model Comparisons

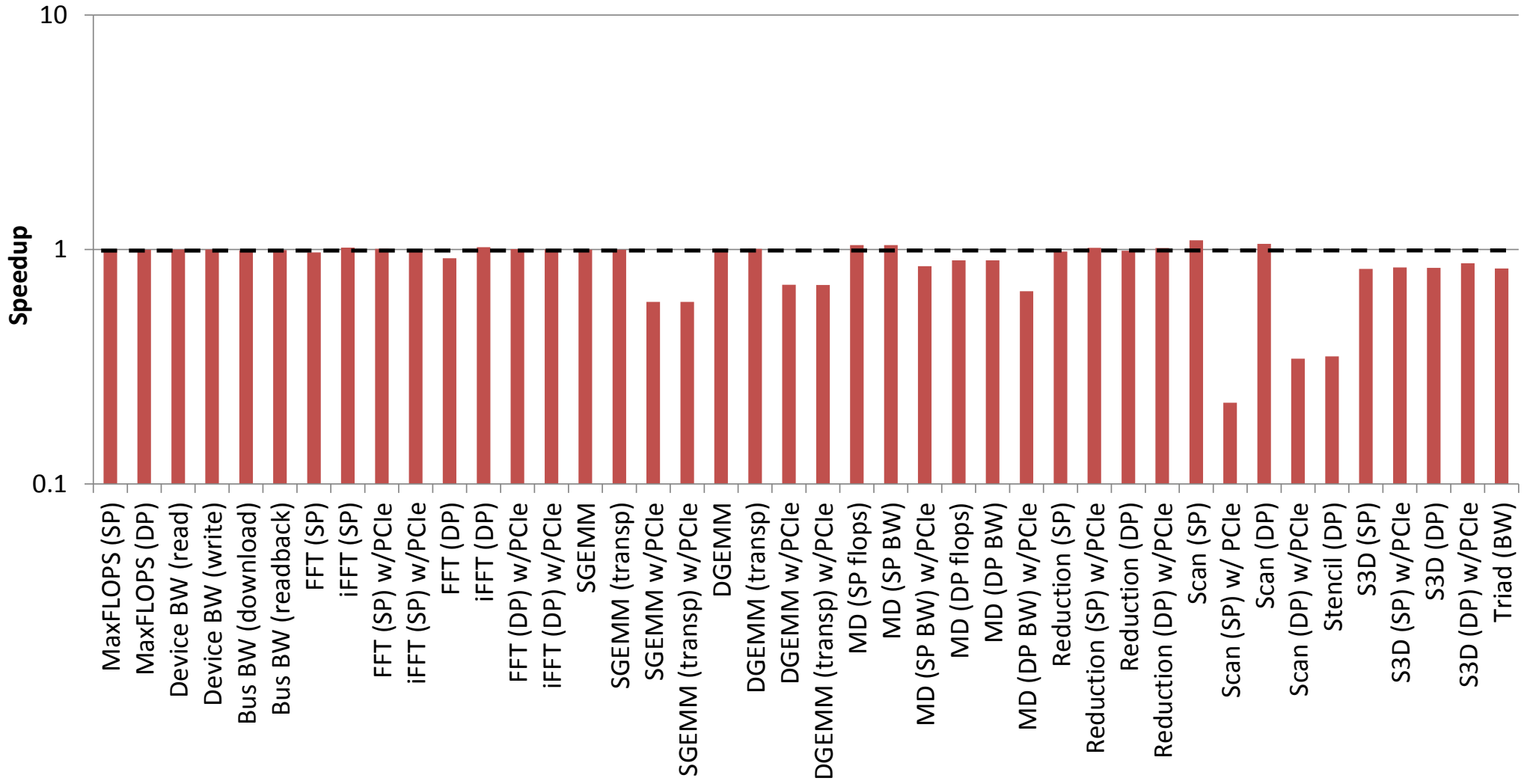
- Different explicit models
  - CUDA vs OpenCL was a big interest for SHOC 1.0
- Native versus offload models within a device
  - Xeon Phi with OpenMP
- Generational improvements/regressions in APIs/compilers
  - OpenACC and OpenMP+LEO
- Explicit models vs directive models
  - OpenACC vs CUDA
  - OpenMP vs OpenCL

Native vs Offload (Xeon Phi)



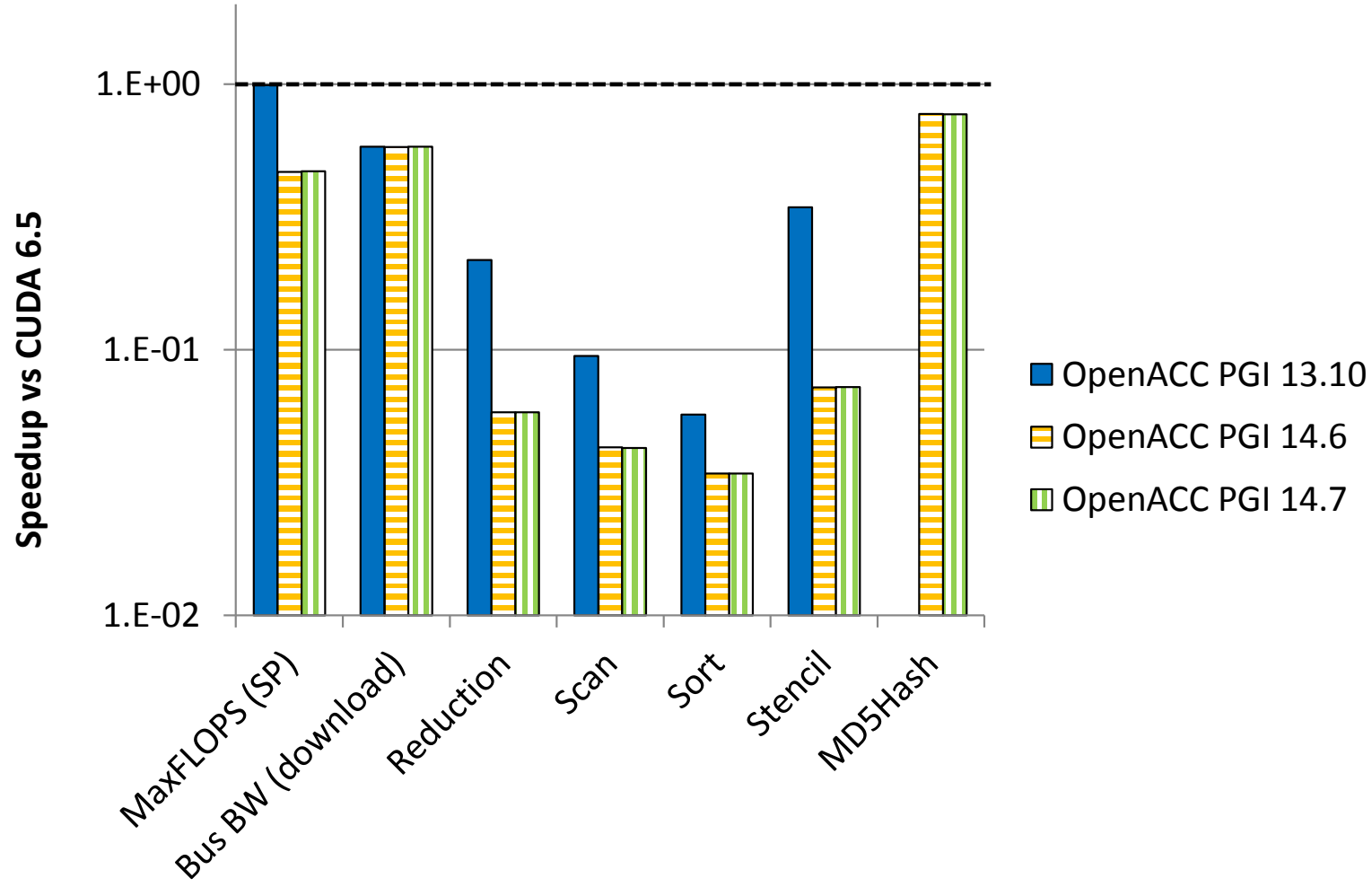
- Benchmarks with PCIe show bigger improvement in Native
  - In particular, see Triad BW
- However using same directives (offload) for both modes cause some Native slowdowns

Compiler Improvement/Regression (Intel 15 vs 13)



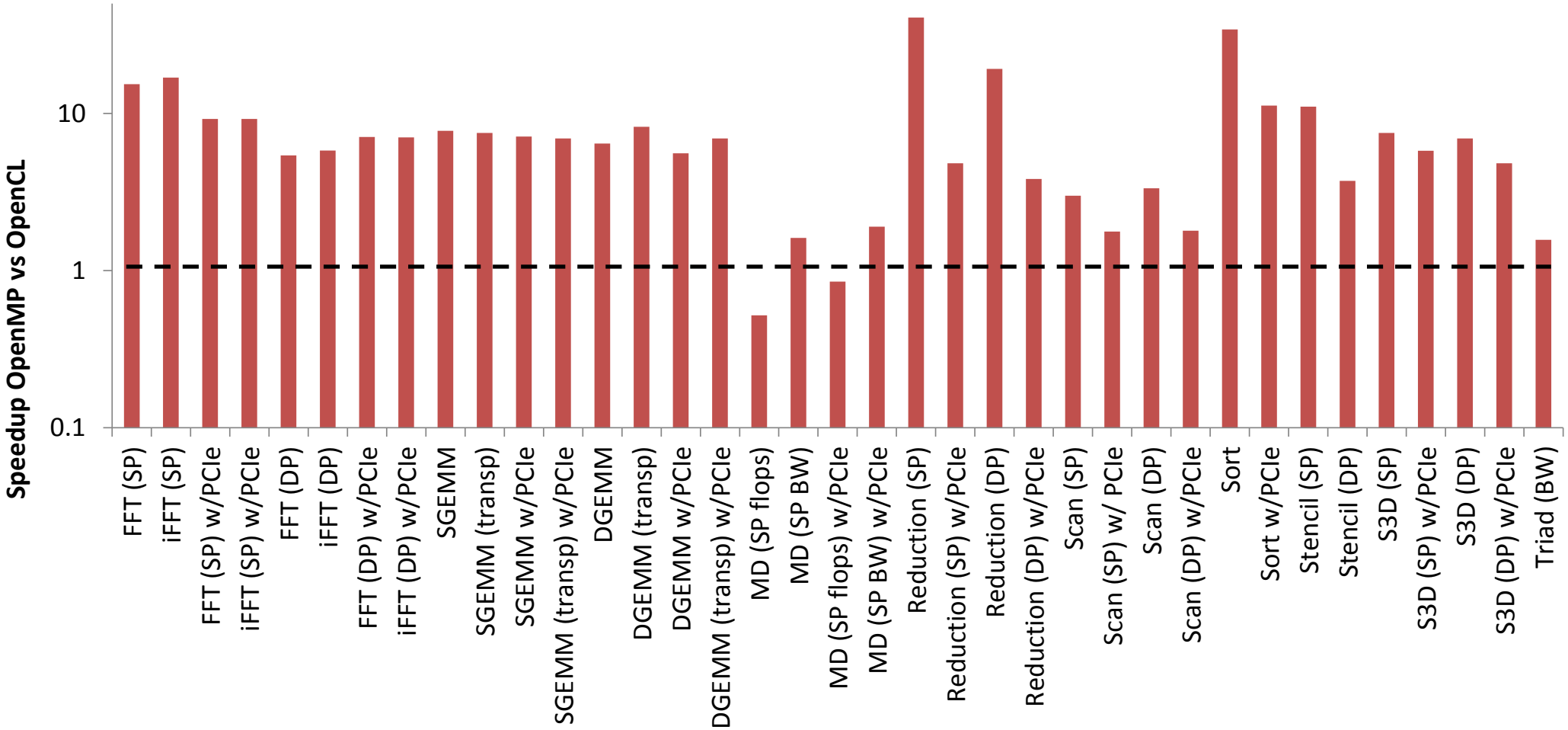
- Improvements were minimal in the newer compiler
- But several major regressions where older compiler was faster

Explicit vs Directive Models (K40 CUDA vs ACC)



- Some OpenACC results approached CUDA results; some were over 10x slower
- Generally saw performance regressions, **not** performance improvements, with newer compiler
  - except one case when older compiler simply generated incorrect binary

Explicit vs Directive Models (MIC OpenMP vs OpenCL)



- Level 0 results (not shown) were nearly identical
- In these Level 1 & 2 kernels, OpenMP was almost always faster than OpenCL

Conclusion

OAK RIDGE National Laboratory

SHOC is useful for benchmarking these systems

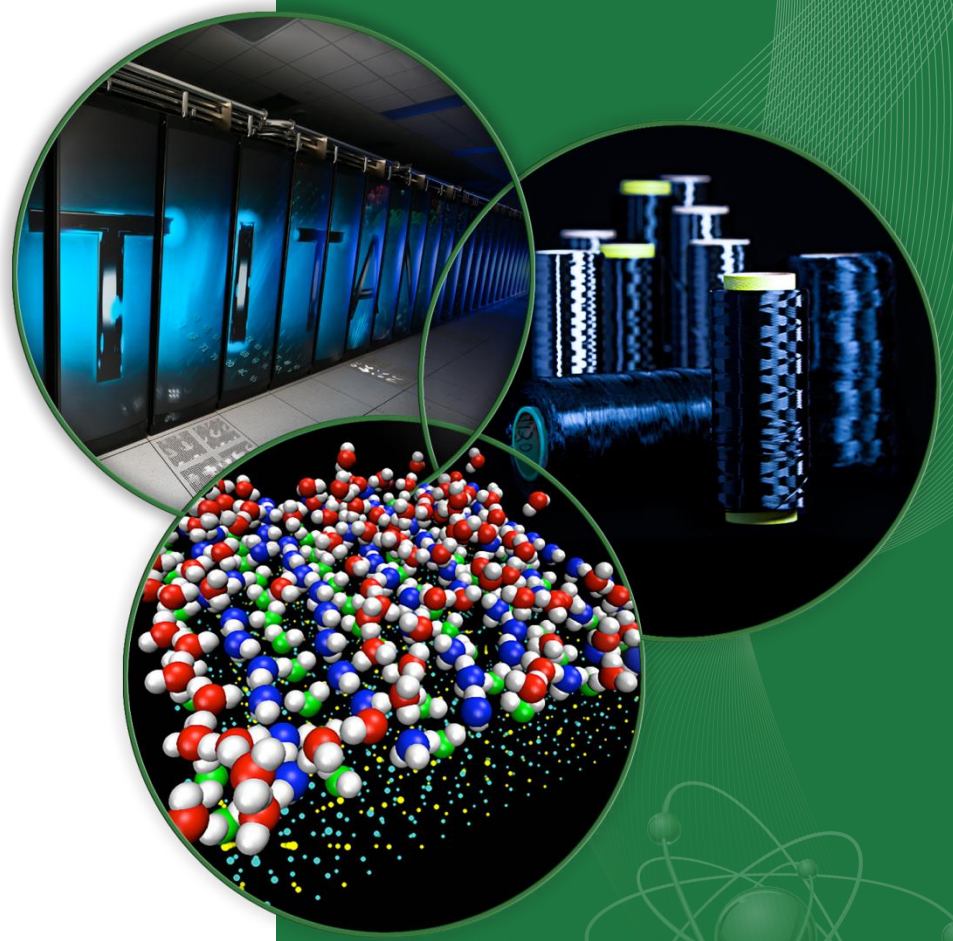
- Wider variety of kernels in SHOC 2.0
  - allows a broader view of device performance
- Wider variety of programming model support in SHOC 2.0
  - allows a wider array of device support
- Longitudinal studies
  - across software / hardware generations
- Cross-sectional studies
  - across APIs, across device vendors
- Scaling studies
  - device size, device count

Lessons learned in the process

- Compiler directive support not yet mature
  - some bugs, occasional language issues
  - many performance regressions over time
  - minor compilation differences impact performance
- Lack of hardware support hurts performance
  - e.g. shared memory critical for some kernels, difficult to access with directives
  - potentially work around with API-specific primitives or language features
- Directives imply portability, but not performance portability
  - difficult to re-imagine key kernels in directive-centric paradigm

Thanks!

Questions?



ORNL is managed by UT-Battelle
for the US Department of Energy

OAK RIDGE National Laboratory