Prof Simon McIntosh-Smith

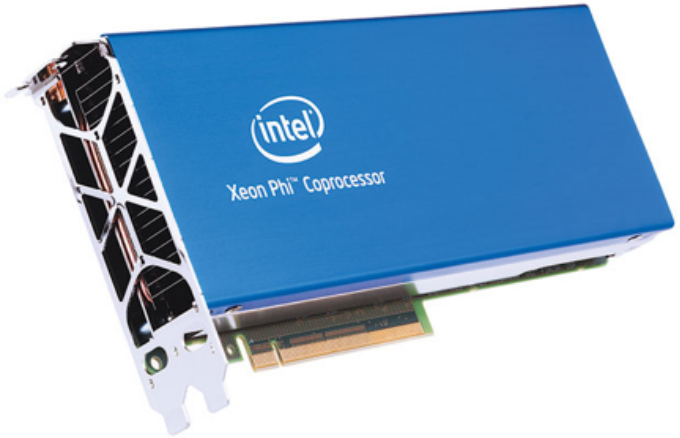
HPC research group

University of Bristol

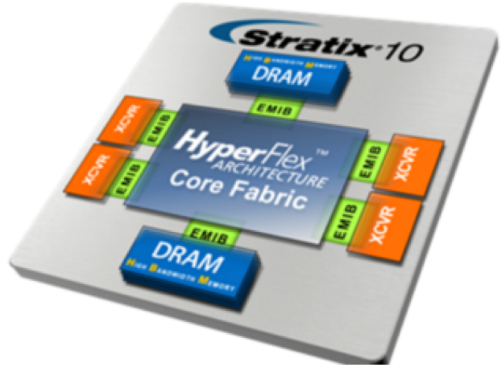
OpenMP on Future Architectures

Recent processor trends in HPC

Many-core CPUs

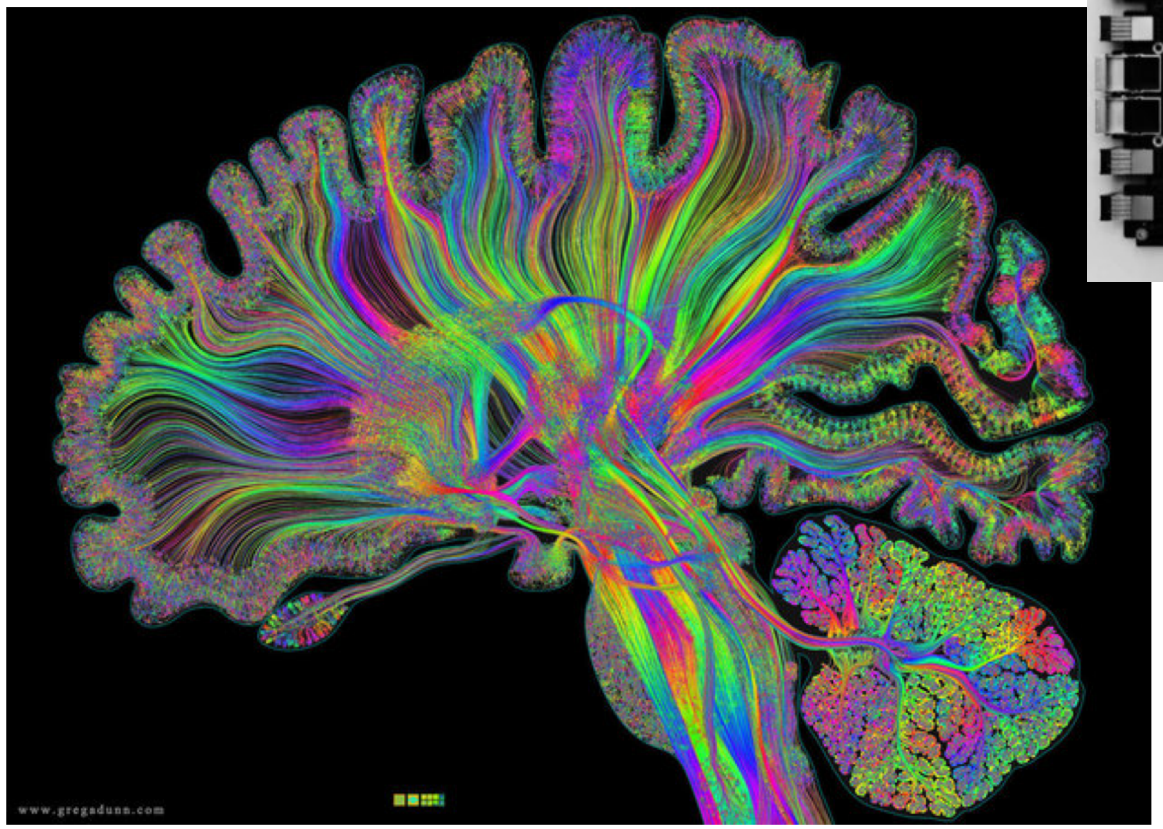


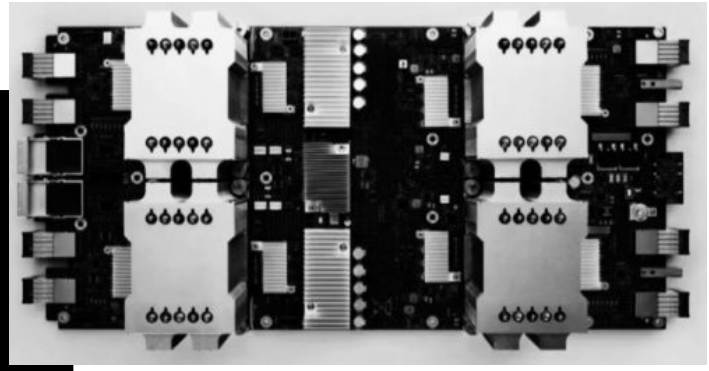
FPGAs

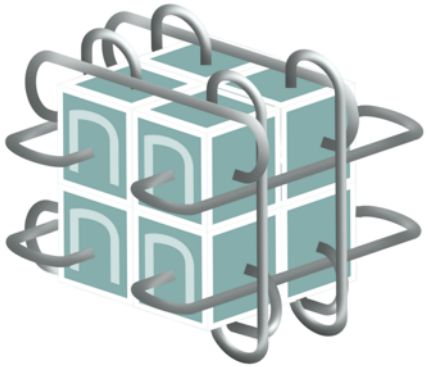


GPUs

Emerging architectures

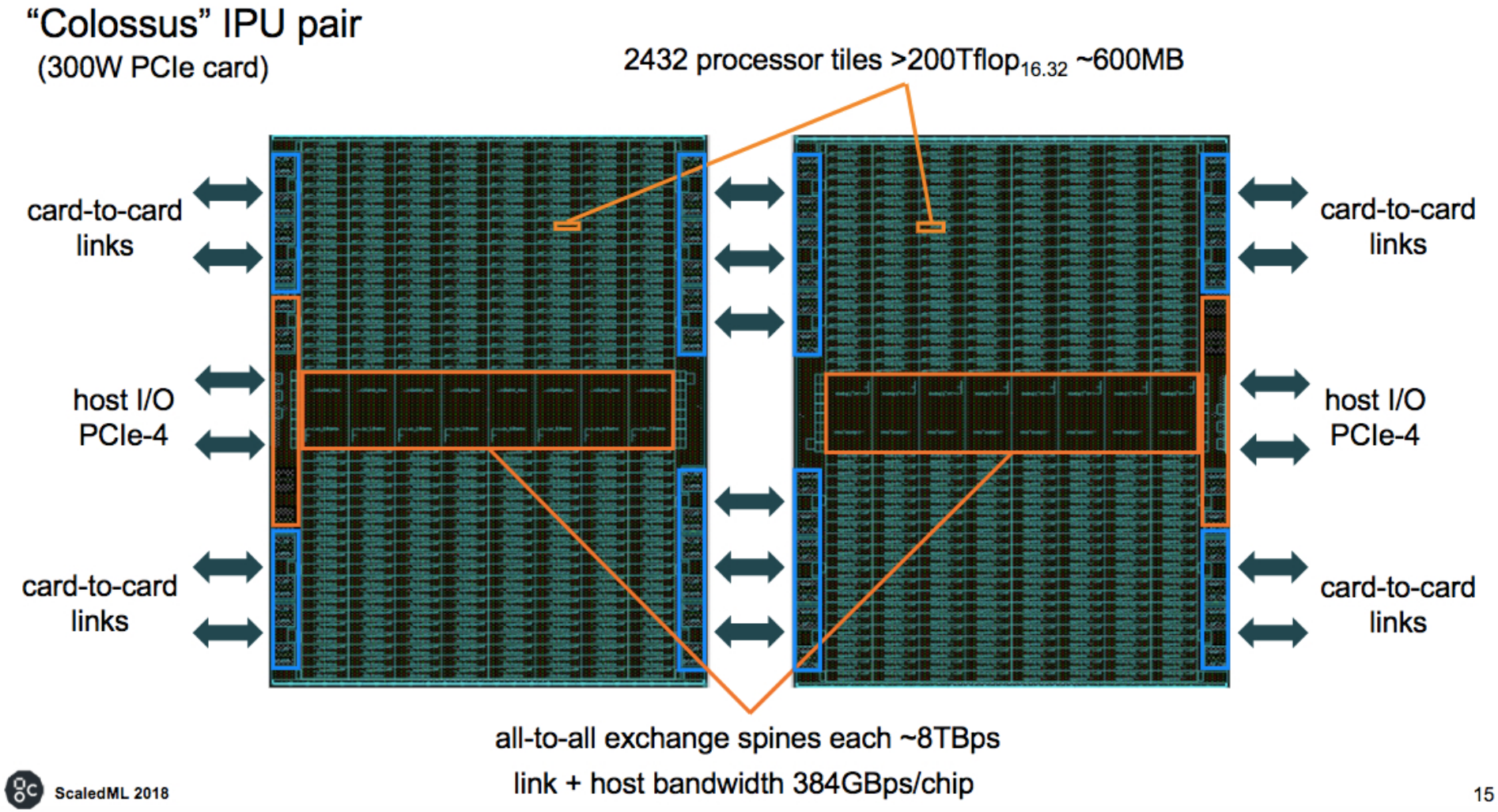






Google's Tensorflow Processing Unit (TPU), GraphCore, Intel's Nervana

GRAPHCORE IPU pair – 600MB @ 90TB/s



University of BRISTOL

http://gw4.ac.uk/isambard/

GW4

Recent CPU trends

- CPUs have evolved to include **lots of cores** and **wide vector units**

- The latest Intel Skylakes have up to 28 cores each
  - 56 cores, 256 GB/s, >3.7 TFLOP/s (dual socket node)
    - Intel Xeon Platinum 8176 (Skylake), 2.1GHz

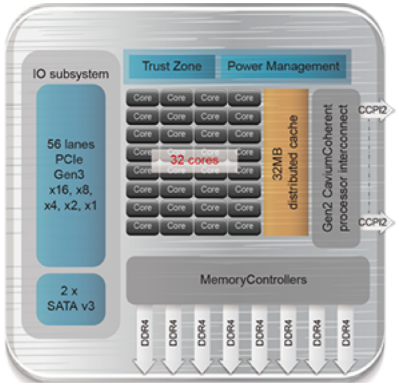
- Starting to see **renewed competition in CPUs**: AMD's EPYC, but especially the emerging Arm server CPU ecosystem

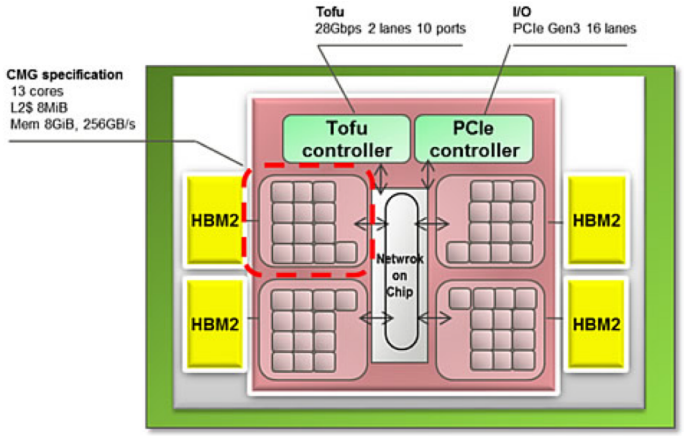
Current Arm server CPU vendors

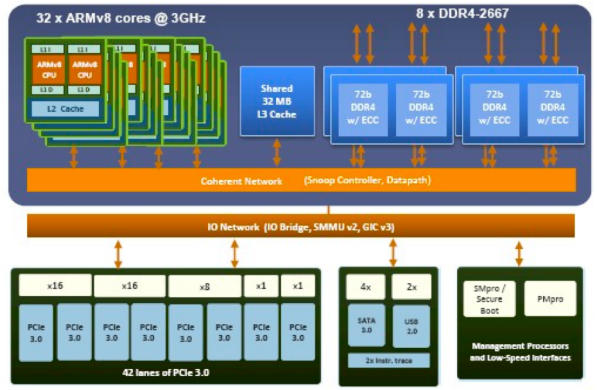
CAVIUM

FUJITSU

AMPERE™







http://gw4.ac.uk/isambard/

University of BRISTOL

GW4

So what can we do?

We can look at how well OpenMP is doing on the most extreme processors that support it today

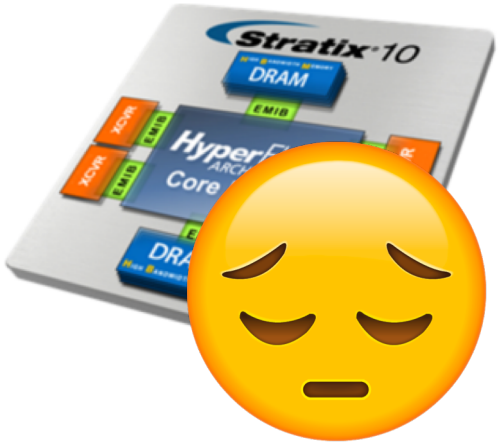
The highest core count CPUs:

- Skylake, ThunderX2, KNL

GPUs:

- P100, V100

FPGAs

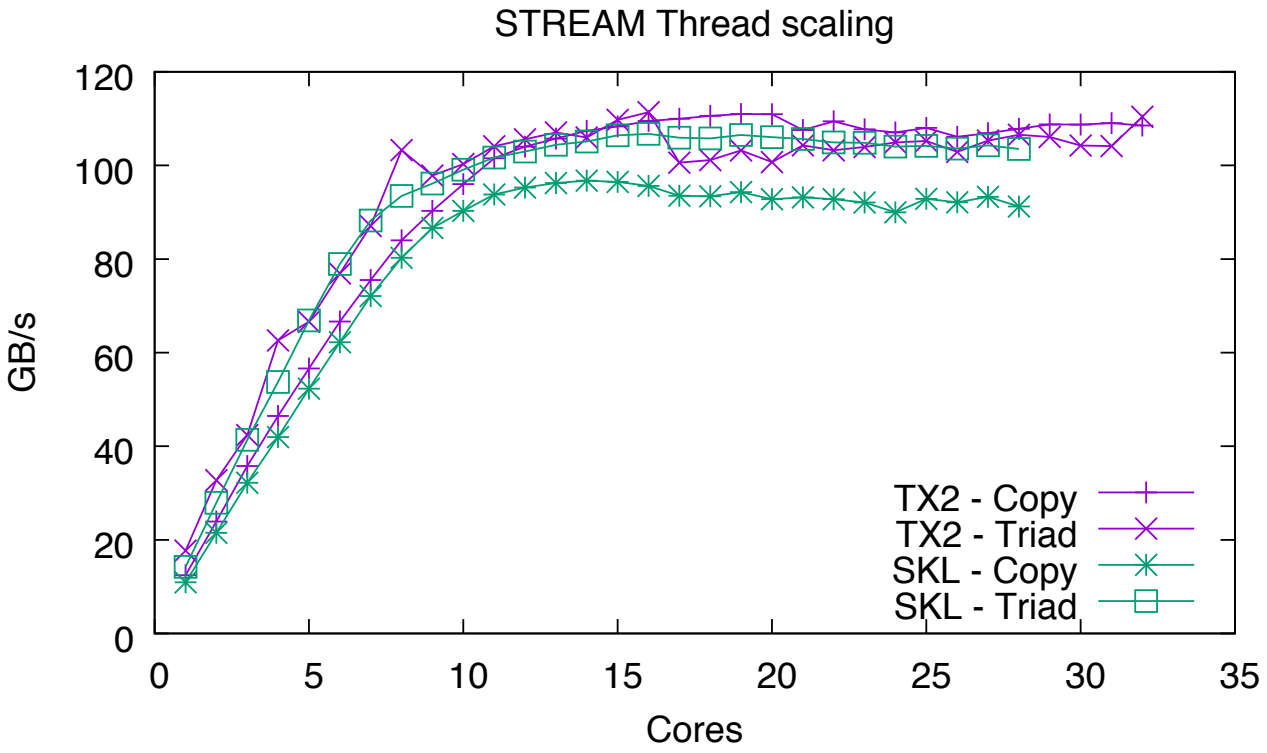


Processor	Cores	Clock speed GHz	TDP Watts	FP64 TFLOP/s	Bandwidth GB/s
Broadwell	2×22	2.2	145	1.55	154
Skylake Gold	2×20	2.4	150	3.07	256
Skylake Platinum	2×28	2.1	165	3.76	256
Knights Landing	1×64	1.3	215	2.66	~490
ThunderX2	2×32	2.2	175	1.13	320

BDW 22c Intel Broadwell E5-2699 v4, **$4,115** each (near top-bin)
SKL 20c Intel Skylake Gold 6148, **$3,078** each
SKL 28c Intel Skylake Platinum 8176, **$8,719** each (near top-bin)
TX2 32c Cavium ThunderX2, **$1,795** each (near top-bin)

All list prices taken from vendors' own websites in May 2018

So how is OpenMP doing on lots of cores?



SKL 28c 2.1GHz, Intel icc 2018

TX2 32c 2.2GHz, underclocked memory (2200MHz vs 2666MHz), GCC 8.1

Both scale well until bandwidth saturated.

SKL does better on triad than on copy (2r1w vs. 1r1w). TX2 is less affected by the difference.

No significant performance penalty for using more threads than were necessary to max out the bandwidth.

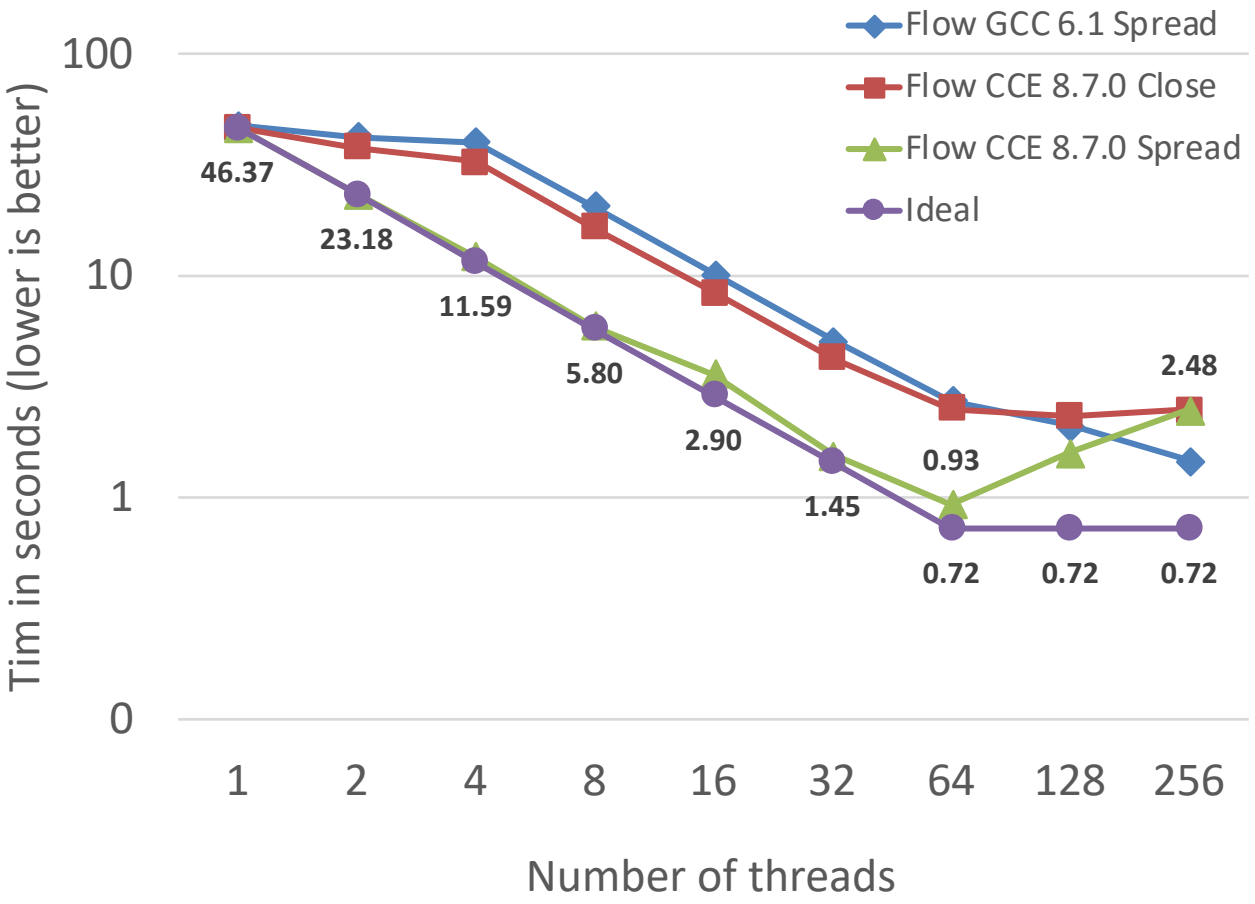
What about on more complex codes?

The HPC research group in Bristol does a lot of work analyzing performance of codes related to high-energy physics.

A lot of this work makes use of mini-apps, such as Cloverleaf (hydrodynamics), TeaLeaf (heat diffusion), and SNAP (deterministic neutral particle transport).

We now have some new mini-apps which make these experiments even easier: **flow** (hydro), **hot** (heat diffusion), and **neutral** (Monte Carlo neutral particle transport).

‘Flow’ hydrodynamics OpenMP scaling on 64c ThunderX2

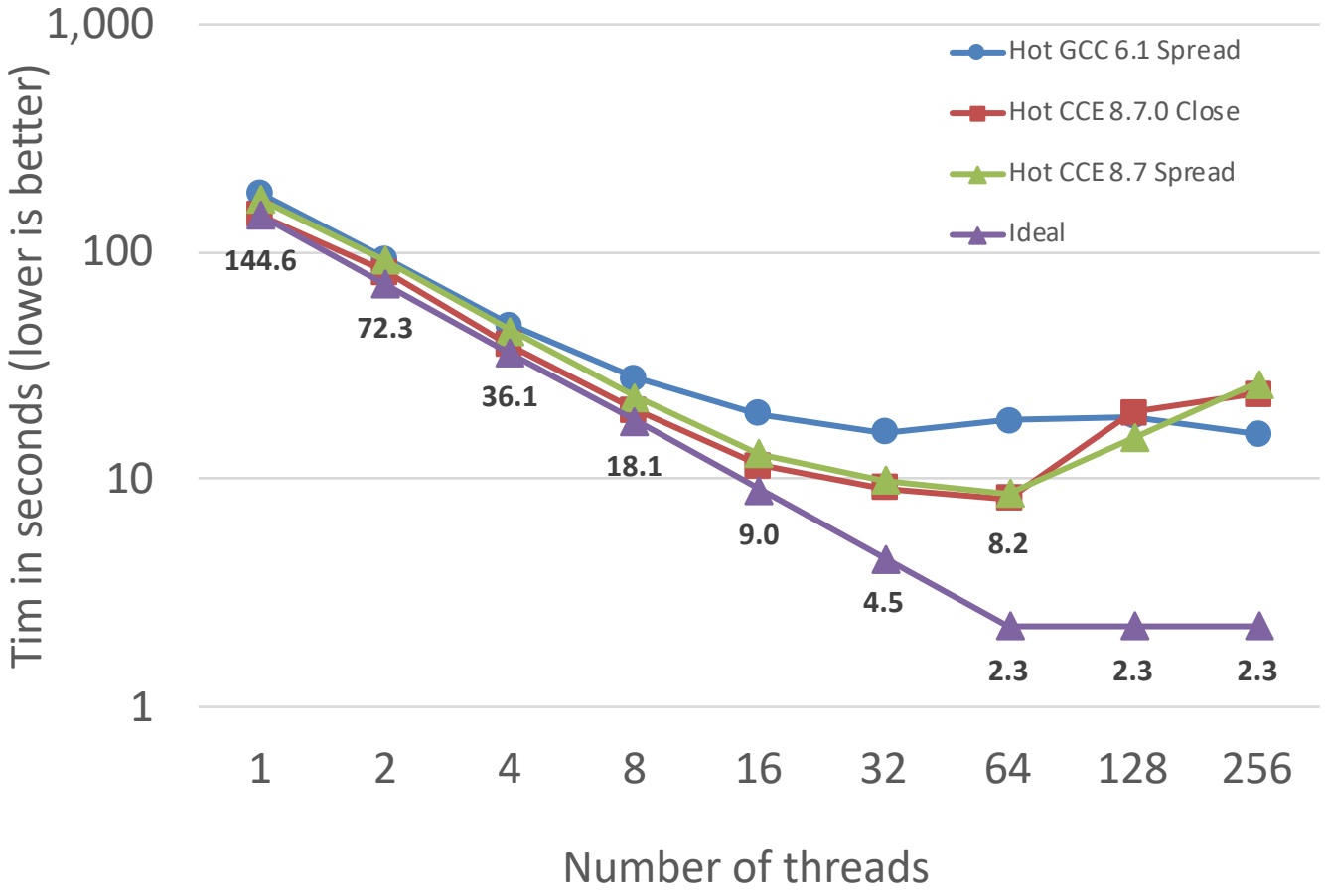


Nearly ideal scaling up to 64 cores.

Adding hyperthreads hurts performance in this instance.

We’d expect this case to scale well.

‘Hot’ heat diffusion OpenMP scaling on 64c ThunderX2

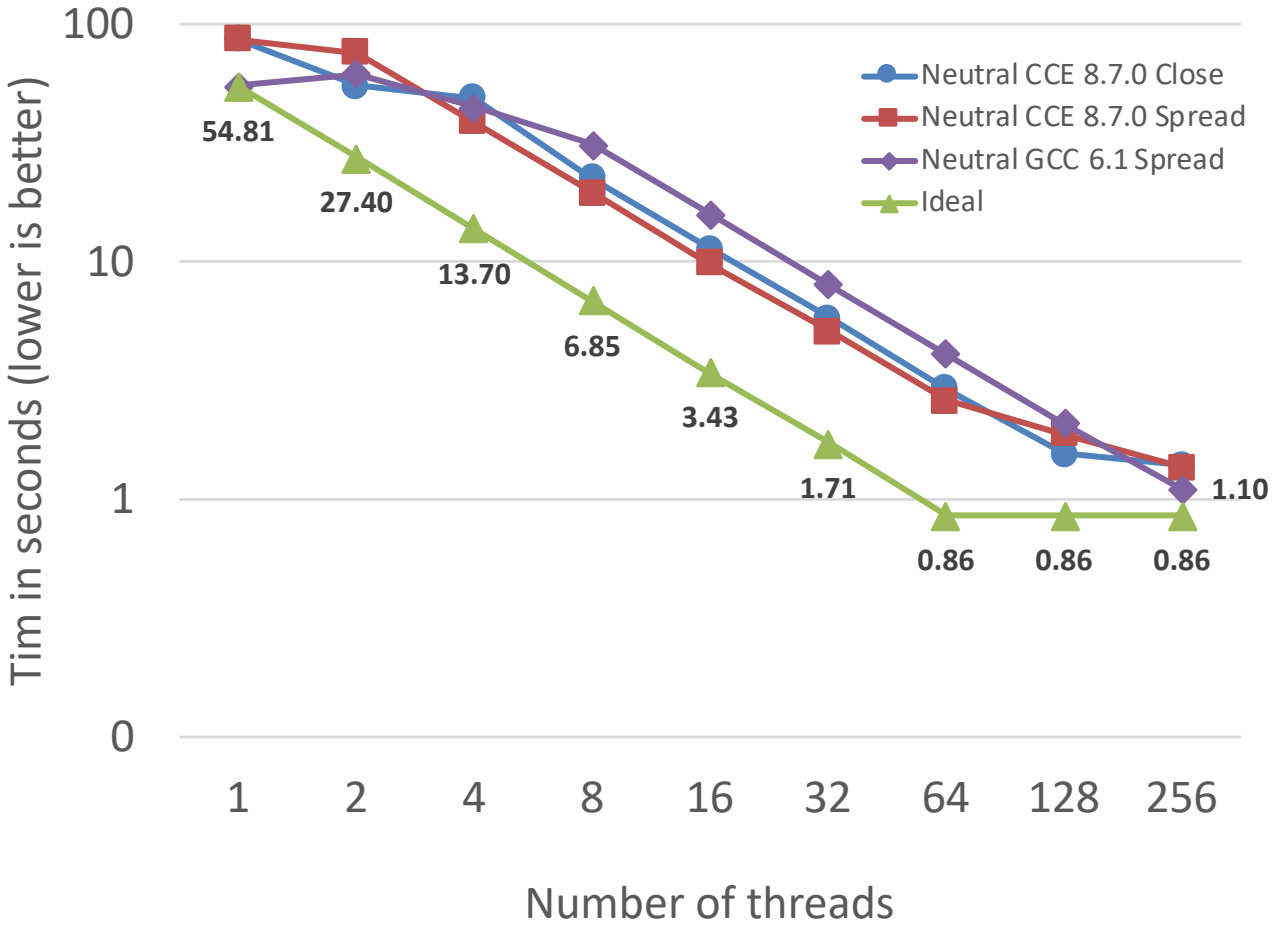


More quickly saturates the memory bandwidth than Flow.

Bandwidth almost saturated by about 16 cores.

Adding hyperthreads hurts performance in this instance too.

‘Neutral’ MC transport OpenMP scaling on 64c ThunderX2

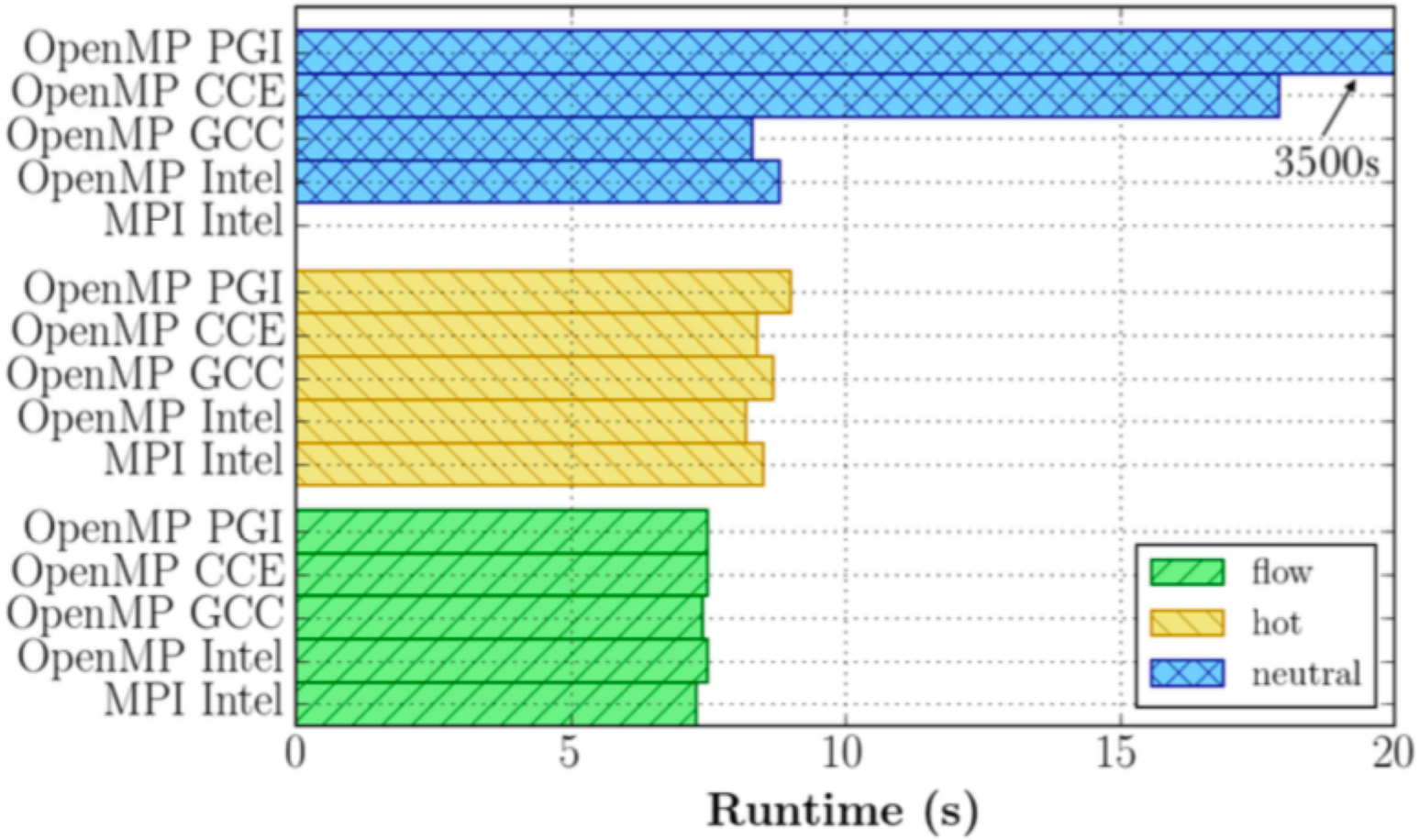


Two kernels:

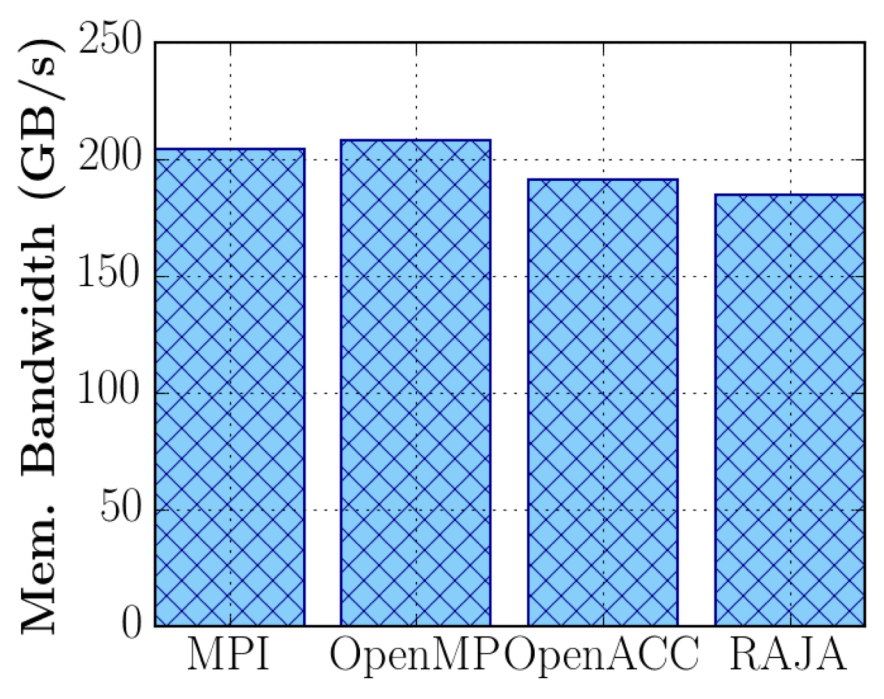
- One compute bound
- One bandwidth bound

Adding hyperthreads helps a lot in this instance, improving performance by a factor of 3.7x with GCC and 1.9x with CCE at 64c.

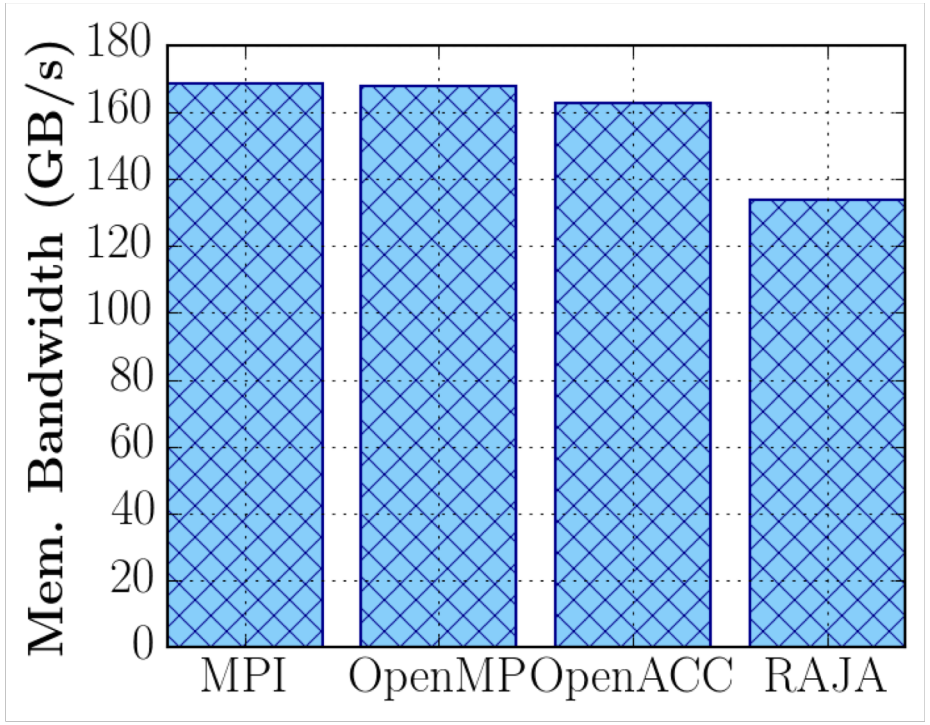
Comparing OpenMP and MPI performance on Skylake



Comparing OpenMP with other parallel languages (SKL)

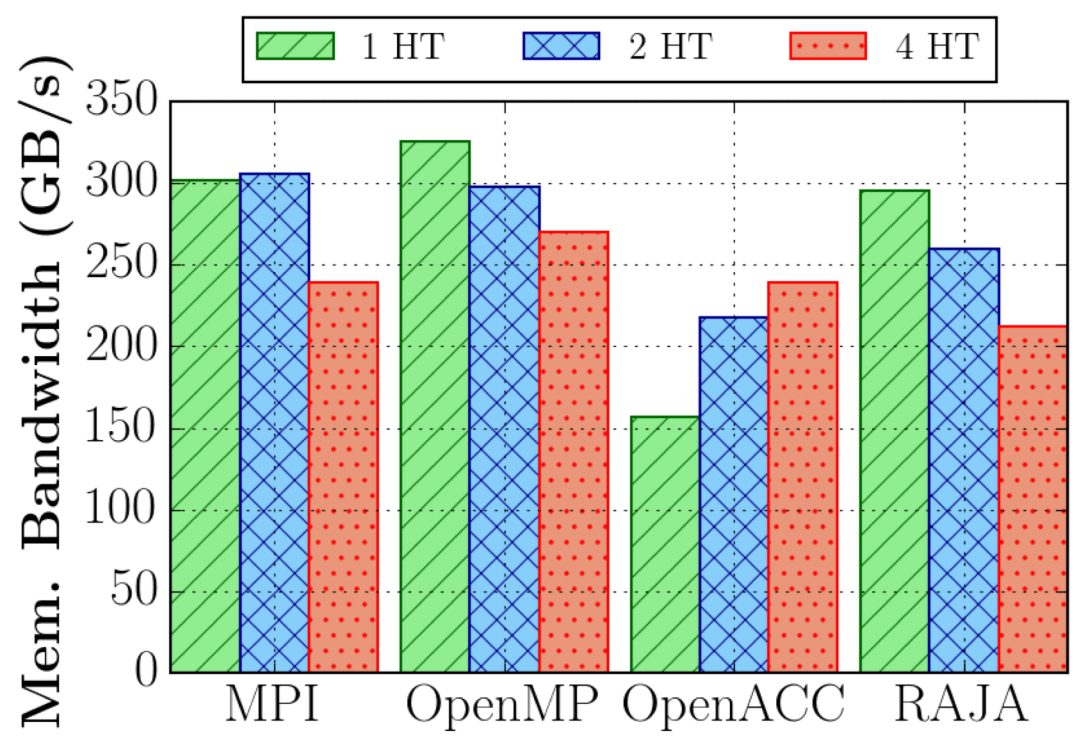


‘Hot’ on Skylake CPU

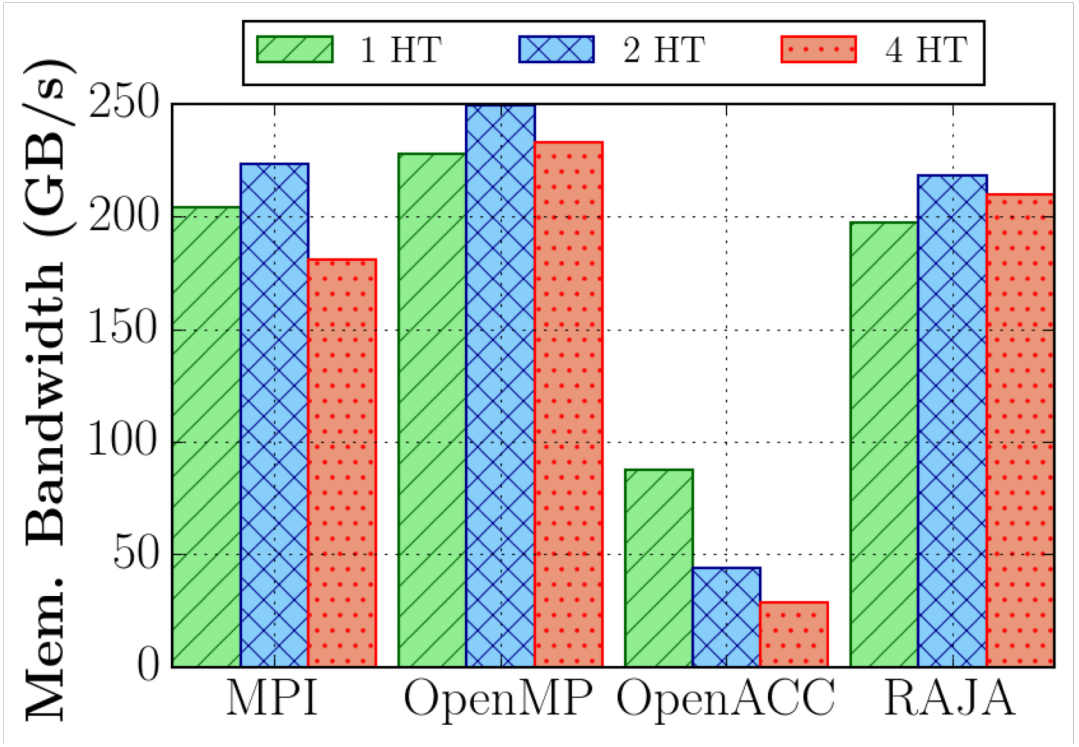


‘Flow’ on Skylake CPU

Comparing OpenMP with other parallel languages (KNL)

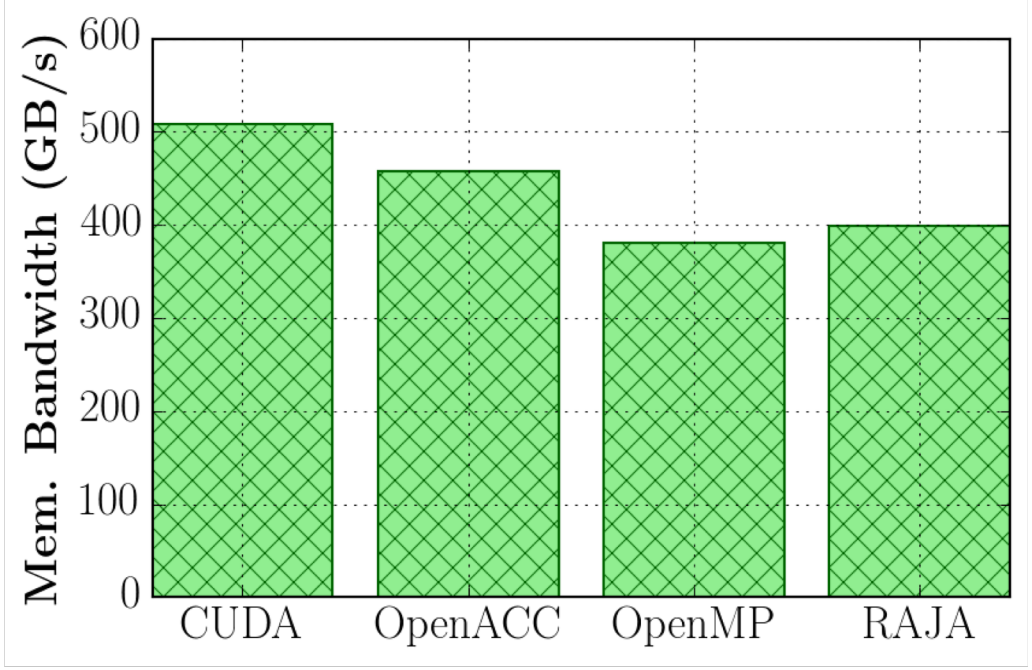


‘Hot’ on KNL CPU

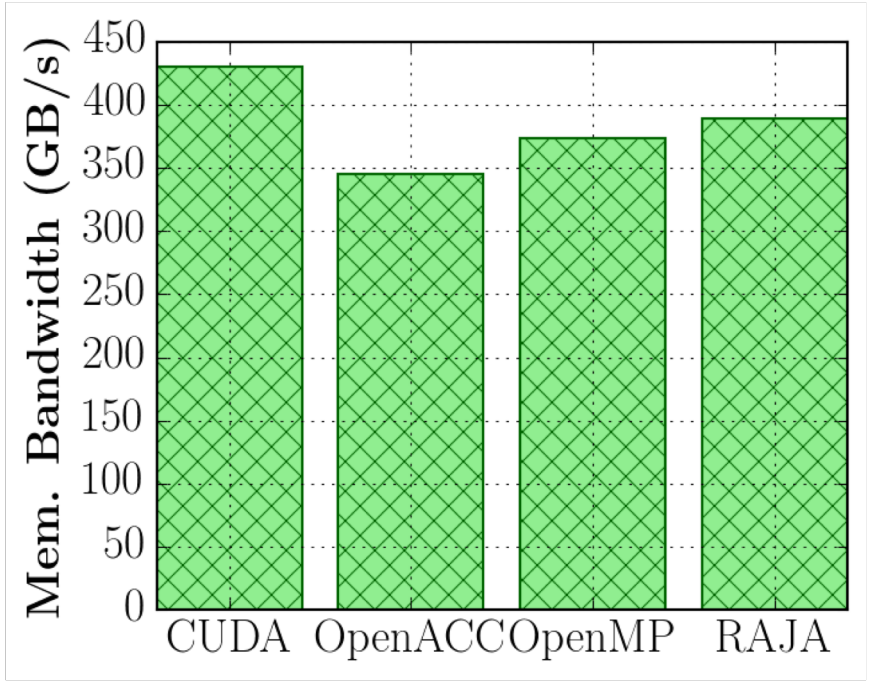


‘Flow’ on KNL CPU

How well is OpenMP doing on GPUs?



‘Hot’ on P100 GPU



‘Flow’ on P100 GPU

OpenMP in production science codes

Benchmark	ThunderX2
STREAM	Pure OpenMP
CloverLeaf	1 MPI per core, OpenMP for 4xSMT
TeaLeaf	1 MPI per core, OpenMP for 2xSMT
SNAP	1 MPI per core, OpenMP for 4xSMT
Neutral	Pure OpenMP
CP2K	1 MPI per core, OpenMP for 2xSMT
GROMACS	1 MPI per core, OpenMP for 4xSMT
NAMD	Charm++ (no MPI or OpenMP)
NEMO	2 MPI per core, no OpenMP
OpenFOAM	4 MPI per core, no OpenMP
OpenSBLI	1 MPI per core, no OpenMP
UM	1 MPI per core, OpenMP for 3xSMT
VASP	1 MPI per core, no OpenMP

https://github.com/UoB-HPC/benchmarks

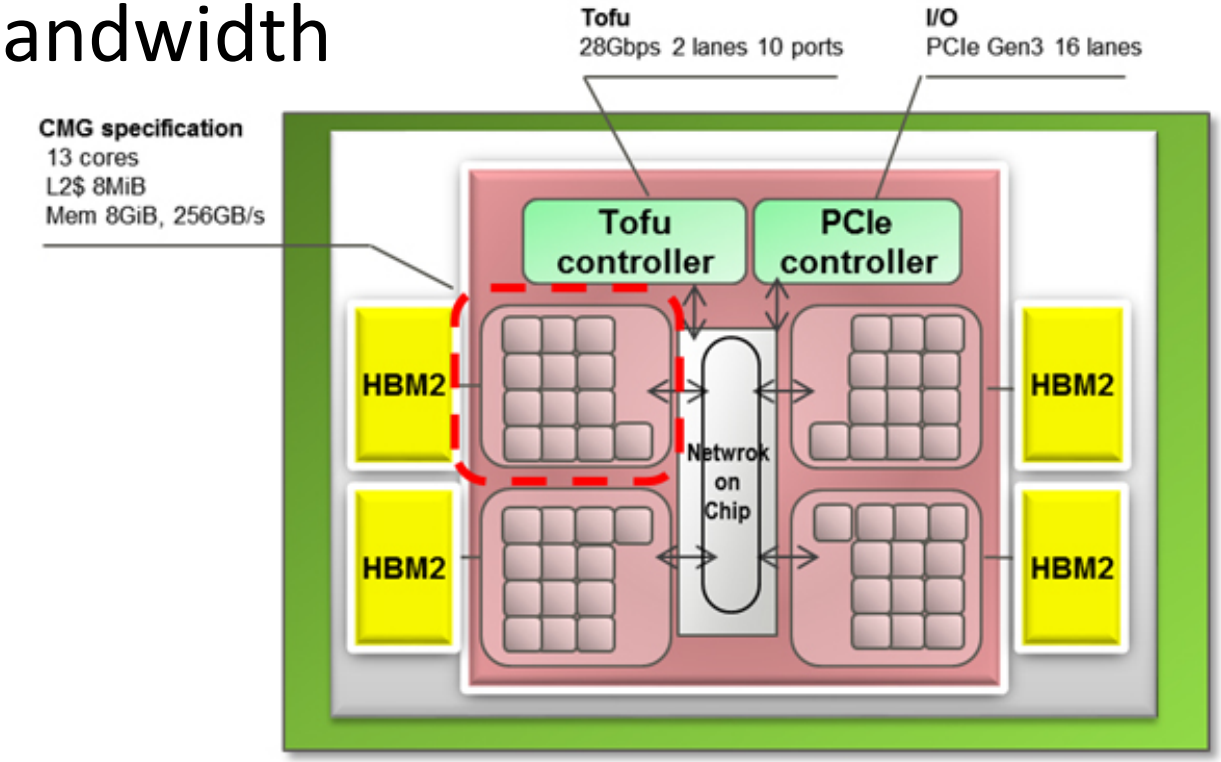
Implications for OpenMP in the future

- More cores
- More data parallelism
- More heterogeneity
- More architectural diversity

- Less memory per core
- Less bandwidth per core

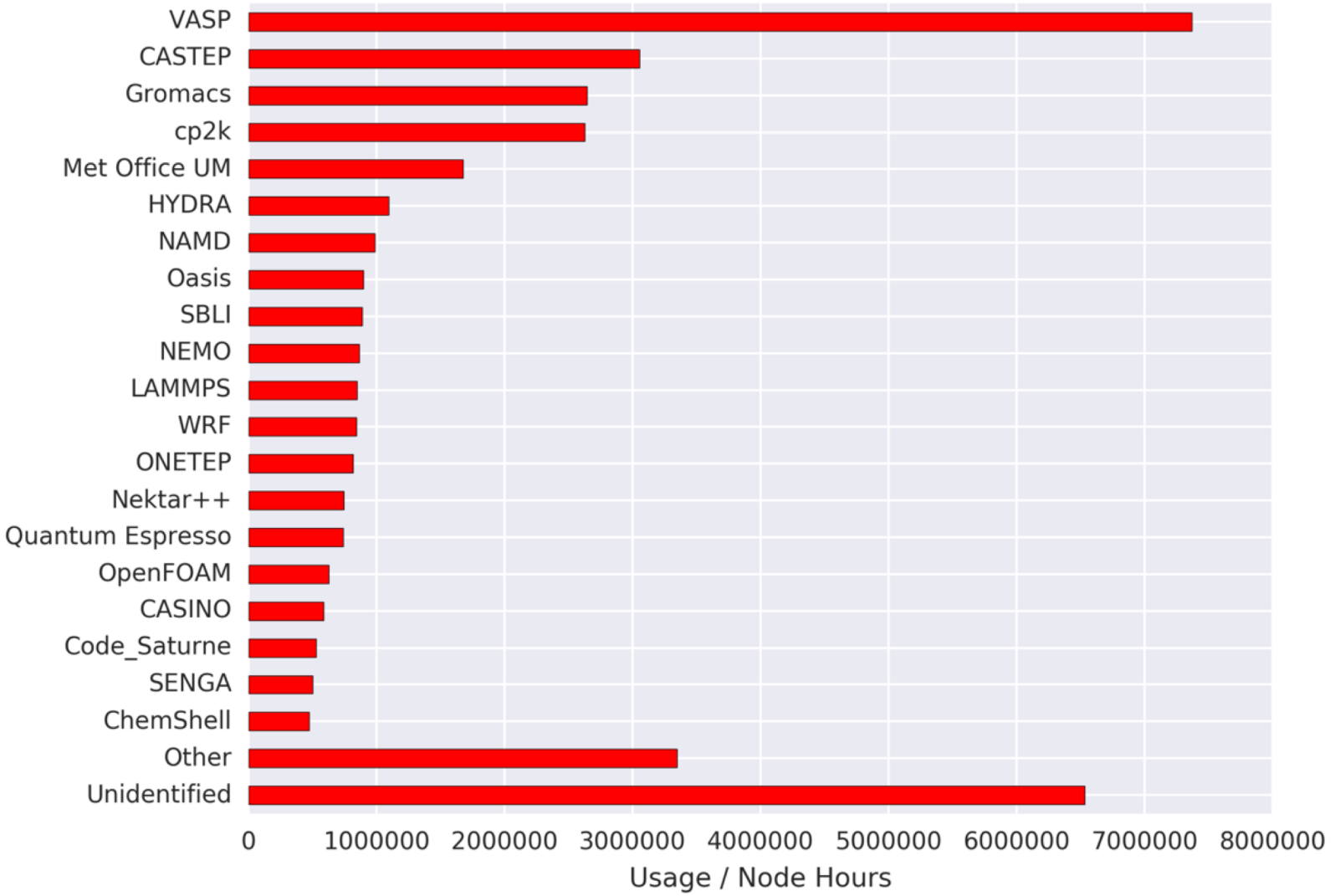
An example forthcoming Arm-based CPU: Fujitsu's A64fx

- 48 cores, no hyperthreading
- 2.7 TFLOP/s double precision, 512-bit vectors (SVE)
- 1 TeraByte/s main memory bandwidth
  - 32 GB HBM2
- ~170 Watts
- High speed interconnect
- First silicon now
- 8.7B transistors, 7nm



http://www.fujitsu.com/global/about/resources/news/press-releases/2018/0822-02.html

The impact of high bandwidth (smaller) memories on OpenMP



Archer usage from a 12 month study.

Archer has 24 IVB cores and 64 GiB per node (2.67GiB/core).

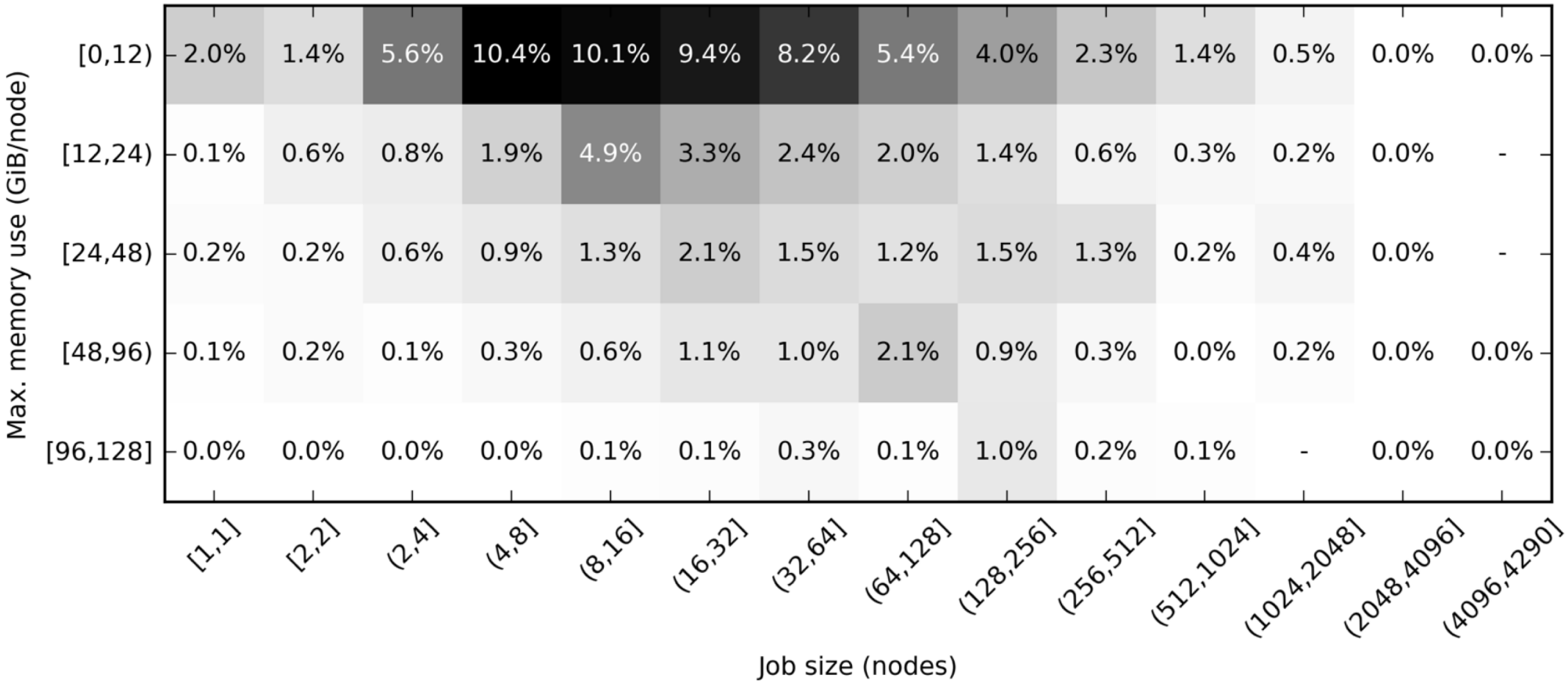
University of BRISTOL

A. Turner and S. McIntosh-Smith. "A survey of application memory usage on a national supercomputer: an analysis of memory requirements on ARCHER." In PMBS, IEEE/ACM SuperComputing 2017.

GW4

Future opportunities: HBM, how much would we need?

Fujitsu's "Post-K" A64fx CPU has 32GB HBM2 for 48 cores, 0.67GB/core

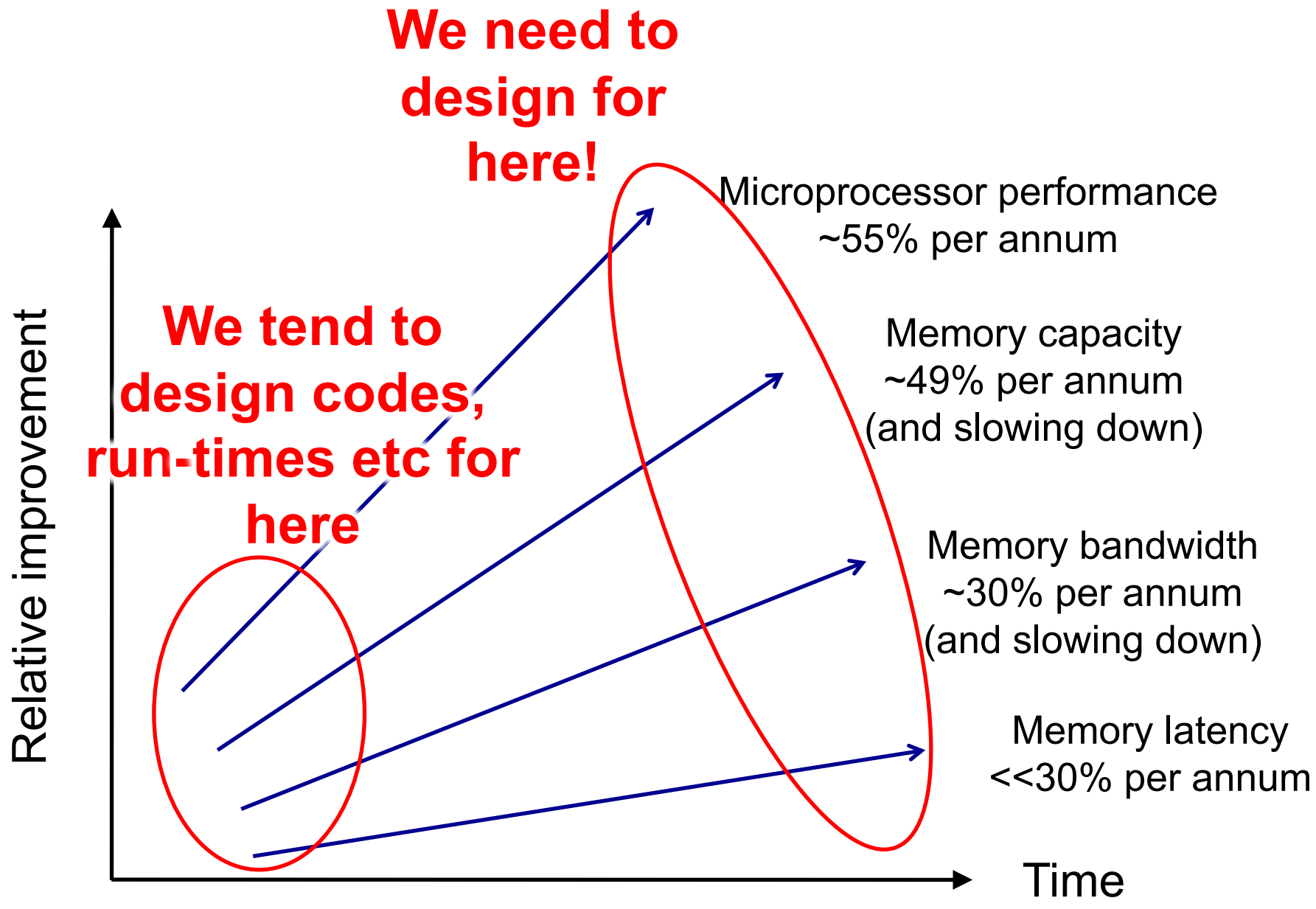


GiB/core	Σ
[0-0.5)	65.4%
[0.5-1.0)	86.8%
[1.0-2.0)	96.2%
[2.0-4.0)	99.9%
[4.0-8.0]	100.0%

University of BRISTOL

A. Turner and S. McIntosh-Smith. "A survey of application memory usage on a national supercomputer: an analysis of memory requirements on ARCHER." In PMBS, IEEE/ACM SuperComputing 2017.

GW4


We need to design for here!
We tend to design codes, run-times etc for here
Microprocessor performance
~55% per annum
Memory capacity
~49% per annum
(and slowing down)
Memory bandwidth
~30% per annum
(and slowing down)
Memory latency
<<30% per annum
Relative improvement
Time

Conclusions

- OpenMP is **scaling well to 64 core systems today**, across multiple compilers
  - Ready for 96 – 128 core systems tomorrow?
- OpenMP on GPUs is **performance competitive with CUDA, OpenACC** et al
  - But **no room for complacency**
- New architectures are often application-optimised, and used via APIs and libraries rather than general-purpose programming languages
  - Will it ever be relevant for a Google TPU to run OpenMP? Do we need to care?
- The threat to OpenMP is being out-competed by MPI in existing codes, and parallel dialects of C++ (Kokkos/Raja etc) on new codes

For more information

Comparative Benchmarking of the First Generation of HPC-Optimised Arm Processors on Isambard

S. McIntosh-Smith, J. Price, T. Deakin and A. Poenaru, CUG 2018, Stockholm

http://uob-hpc.github.io/2018/05/23/CUG18.html

Bristol HPC group: https://uob-hpc.github.io/

Isambard: http://gw4.ac.uk/isambard/

Build and run scripts: https://github.com/UoB-HPC/benchmarks

University of BRISTOL

GW4

- **On the performance portability of structured grid codes on many-core computer architectures**
  S.N. McIntosh-Smith, M. Boulton, D. Curran, & J.R. Price
  ISC, Leipzig, June 2014. DOI: 10.1007/978-3-319-07518-1_4
- **Assessing the Performance Portability of Modern Parallel Programming Models using TeaLeaf**
  Martineau, M., McIntosh-Smith, S. & Gaudin, W.
  Concurrency and Computation: Practice and Experience (Apr 2016)
- **GPU-STREAM v2.0: Benchmarking the achievable memory bandwidth of many-core processors across diverse parallel programming models**
  Deakin, T. J., Price, J., Martineau, M. J. & McIntosh-Smith, S. N.
  First International Workshop on Performance Portable Programming Models for Accelerators (P3MA), ISC 2016
- **The Productivity, Portability and Performance of OpenMP 4.5 for Scientific Applications Targeting Intel CPUs, IBM CPUs, and NVIDIA GPUs**
  M. Martineau and S. McIntosh-Smith, IWOMP 2017, Stony Brook, USA