Performance Benchmarks and Tuning on QB3

Le Yan

Objectives

- Show application performance benchmarks on QB3
- Show methods that may improve the performance

Disclaimers

- Target audience: users who run (and sometimes compile) applications developed by others
  - Not an in-depth guide for programmers and developers

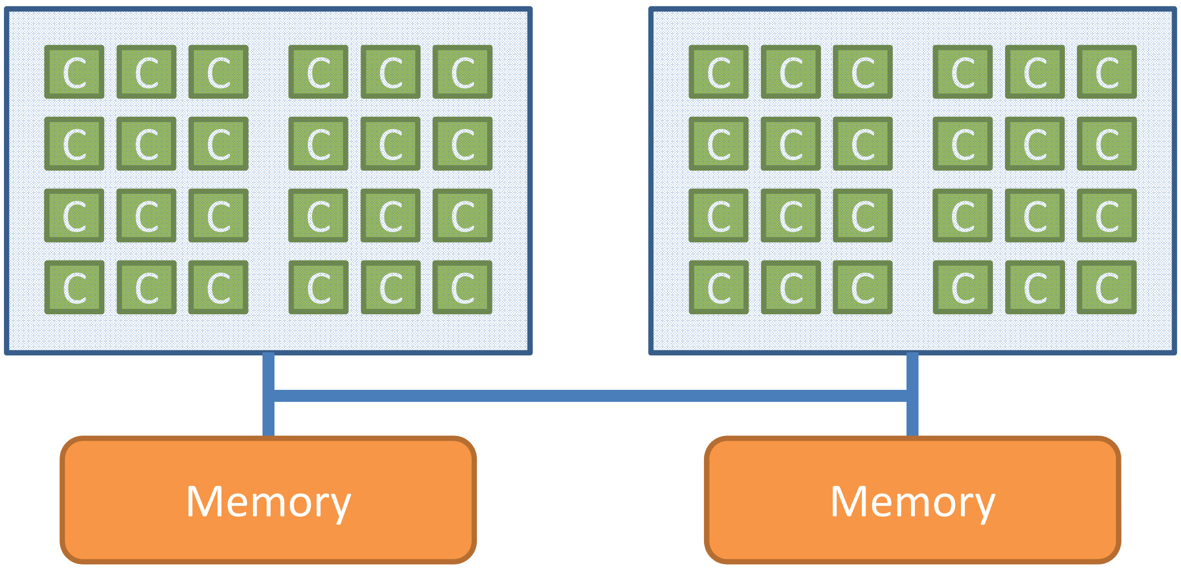
Outline

- Summary of QB3 architecture
- Single node performance
- Multi-node performance

Summary of QB3 architecture

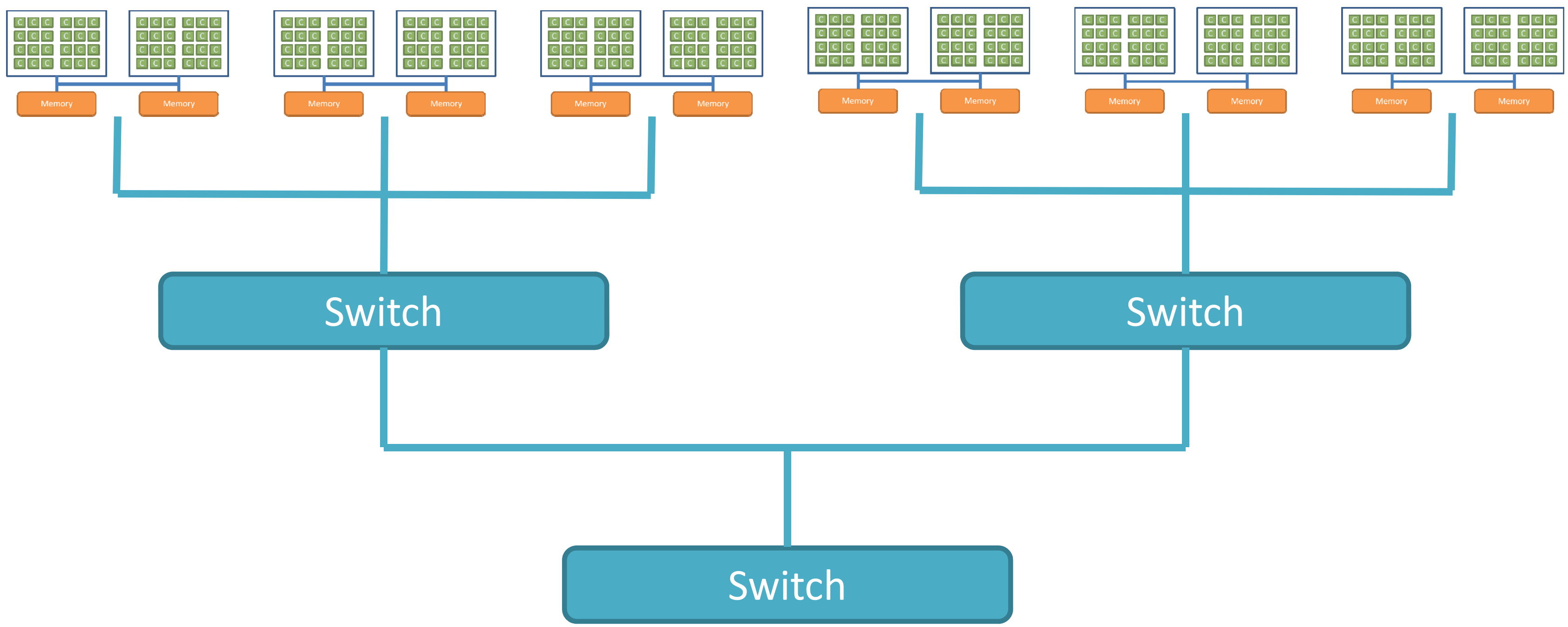
QB3 Architecture

Node level view

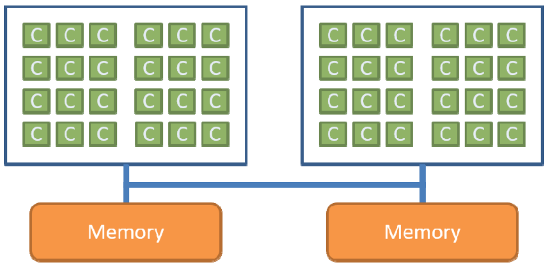


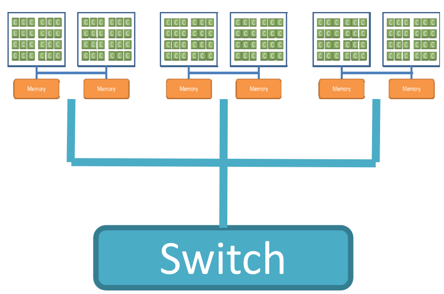
QB3 Architecture

Cluster level view



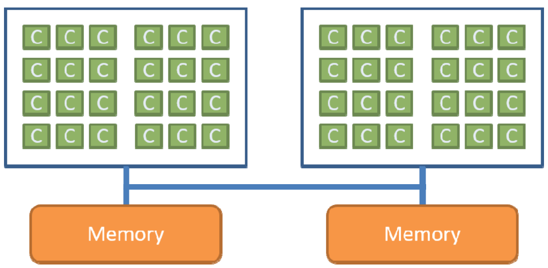
Parallel Paradigms

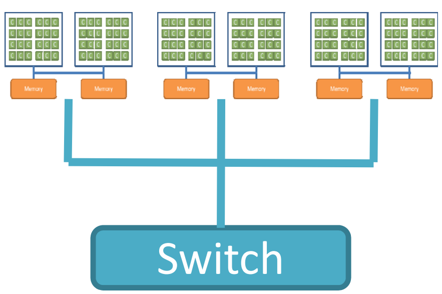




	Pro	Con
Intranode	- Low latency, high bandwidth - Implicit communication - Fine granularity - Dynamic load balancing	- Shared memory system only (Limited to one node)
Internode	- Scalability beyond 1 node	- High latency, low bandwidth - Explicit communication - Hard to load balancing

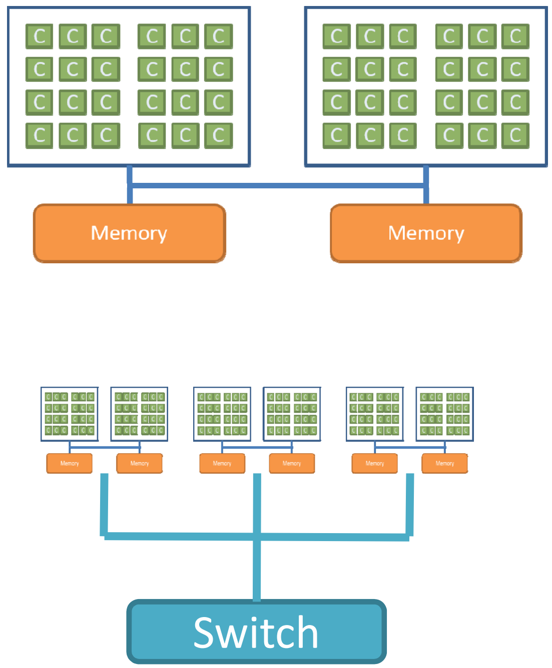
Parallel Paradigms





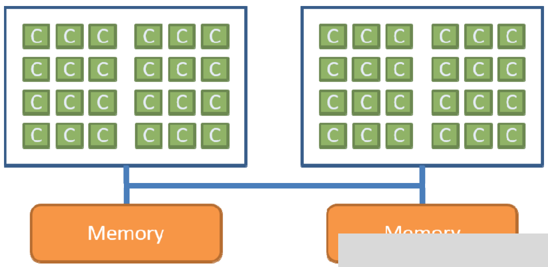
	Pro	Con
Intranode	**OpenMP (Multi-thread)** - Low latency, high bandwidth - Implicit … - Fine granularity - Dynamic load balancing	- Shared memory system …d to one …
Internode	**MPI (Multi-process)** - Scalability beyond …	- High latency, low … …munication - Hard to load balancing

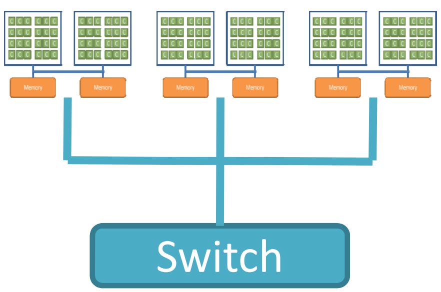
What About MPI+OpenMP Hybrid?



- Getting the benefits from both worlds?
- In theory, yes
- But adding OpenMP to (well-written) MPI programs might hurt the performance
- Hybrid helps to
  – Reduce memory footprint
  – Extend scalability

What About MPI+OpenMP Hybrid?





- Getting the benefits from both worlds?
- In theory, yes
- (well-written) MPI programs might hurt the performance

Your mileage may vary!

- Hybrid helps to
  - Reduce memory footprint
  - Extend scalability

Single Node Performance

QB3 Specification

CPU	Intel Cascade Lake (Xeon Platinum 8260) (2 sockets *24 cores/socket)
CPU frequency	2.4 G Hz
Floating operation per clock cycle (double precision)	32
Memory	192 GB DDR4
Interconnect	Mellanox 100 Gbps Infiniband

QB3 vs QB2 (Node over Node)

	QB3	QB2
CPU frequency	2.4×10^9	2.8×10^9
CPU cores	48	20
Operation per cycle	32	8
Memory bandwidth	~115 GB/s	~48 GB/s
Interconnect	100 Gbps	56 Gbps

<u>Node peak performance</u>

QB3: 48 cores/node * 2.4×10^9 cycles/second * 32 flop/cycle = 3.69×10^{12} flops

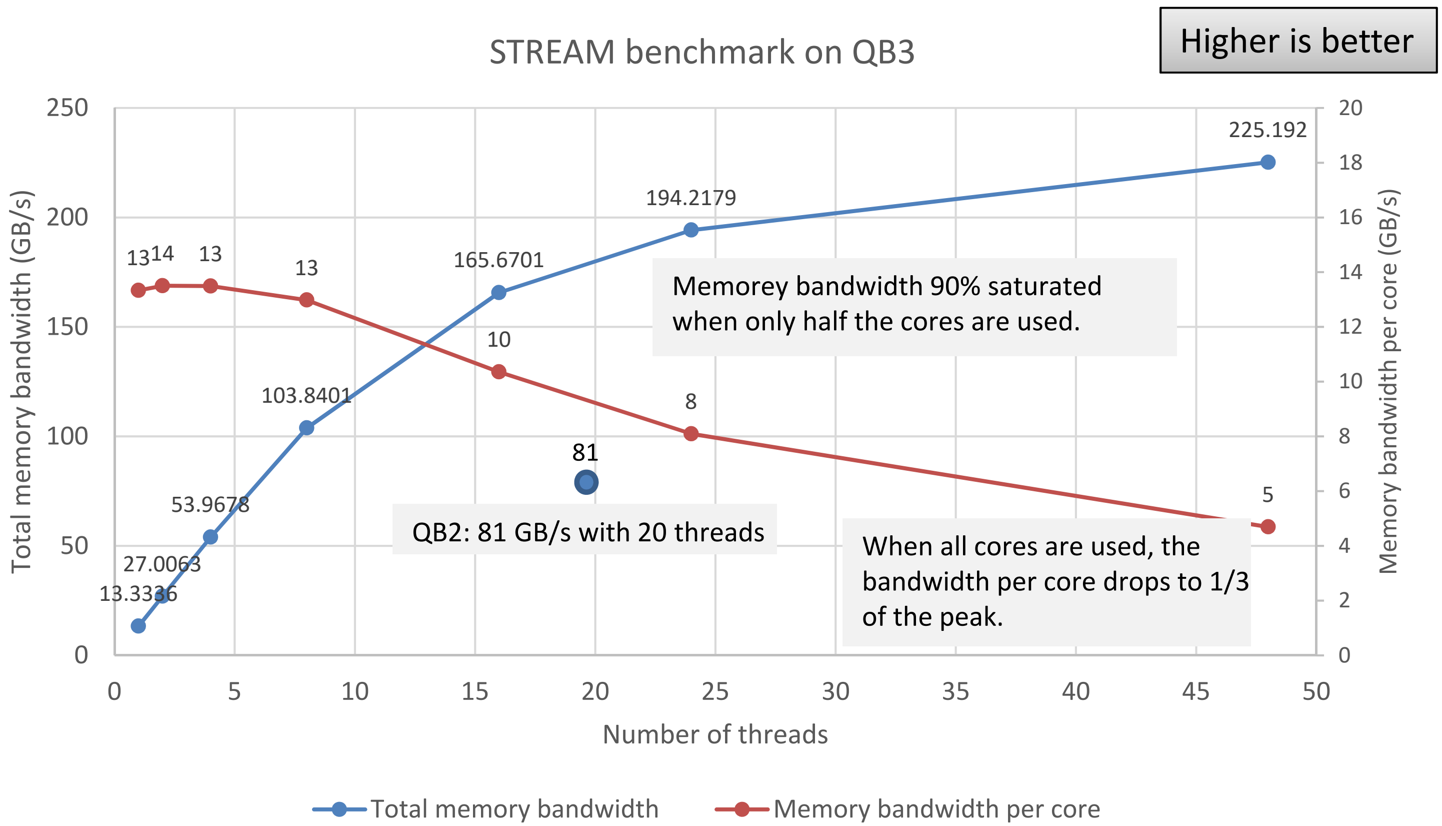
QB2: 20 cores/node * 2.8×10^9 cycles/second * 8 flop/cycle = 0.45×10^{12} flops

Theoretical speedup = 8.2

STREAM Benchmark

- The de facto industry standard benchmark in HPC domain for the measurement of sustainable memory bandwidth (in GB/s).

STREAM Benchmark - Results

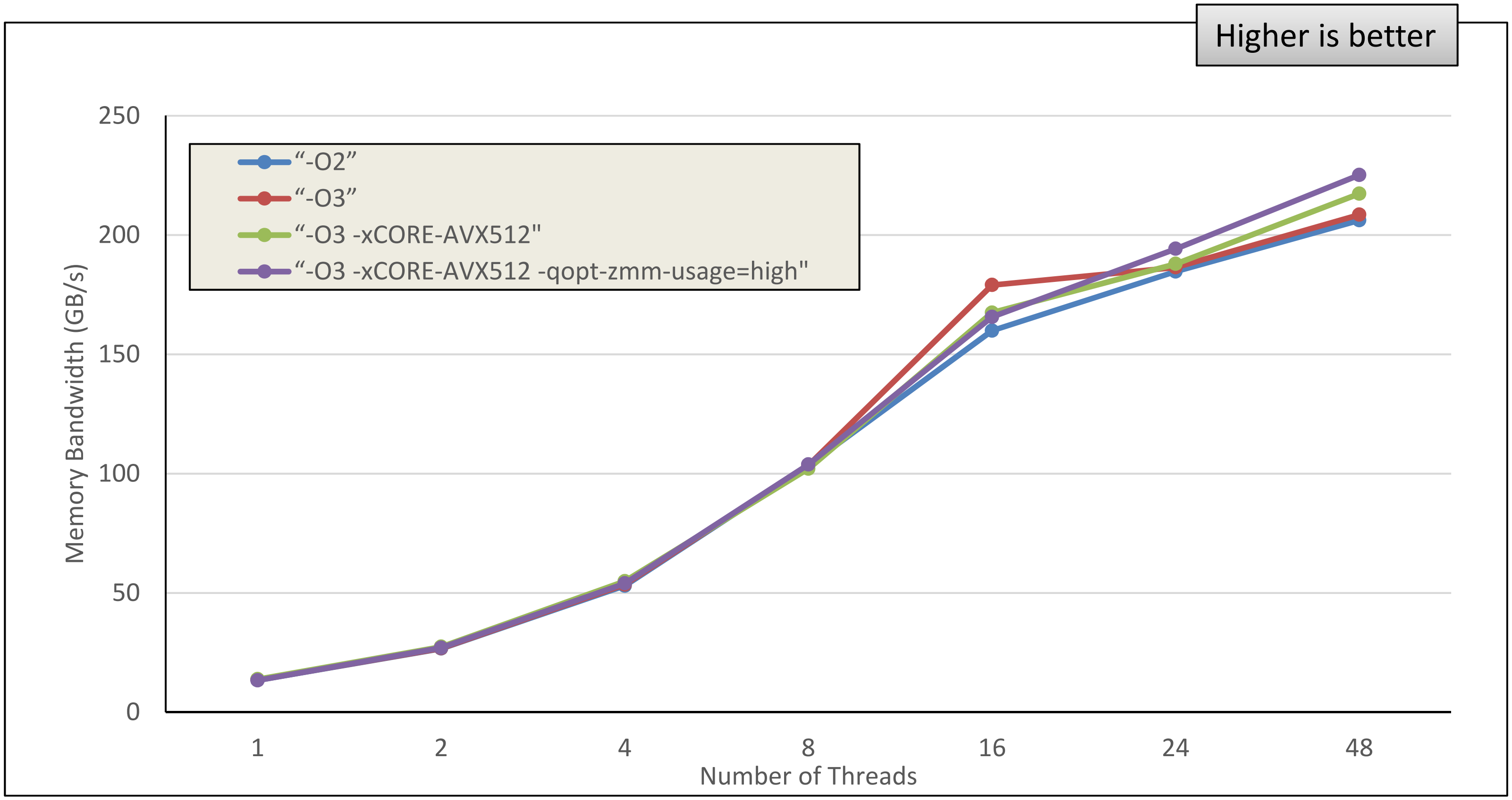


STREAM 5.10
Array size = 4000 MB
Intel 19.0.5 with "-O3 –xCORE-AVX512 -qopt-zmm-usage=high"
Default thread affinity

Compiler Flags (Intel)

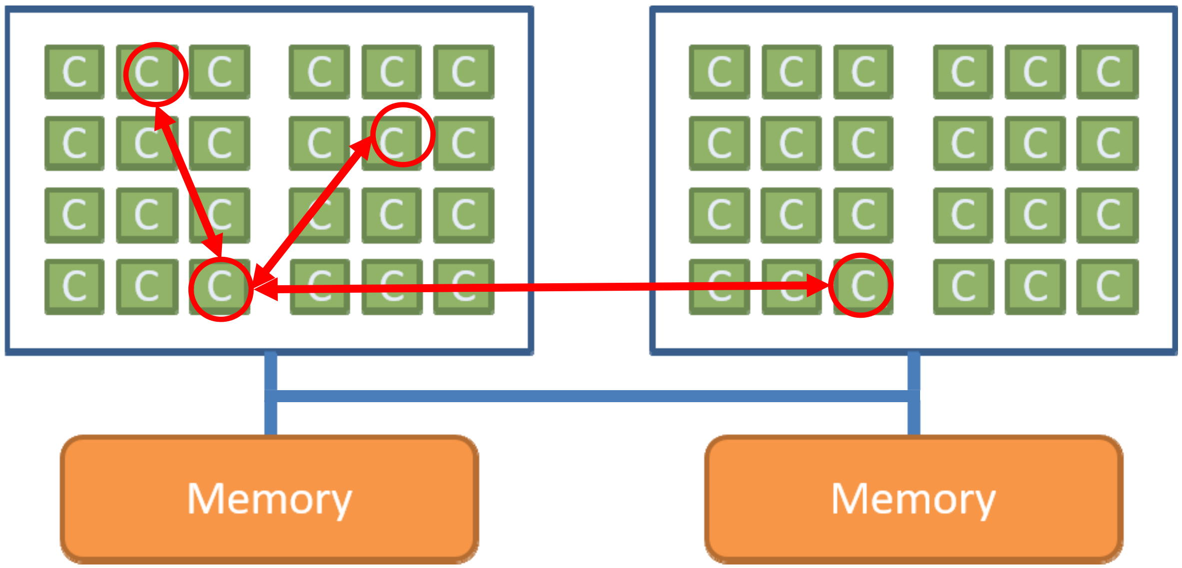
- -O2, -O3
  - Generic, aggregated optimization flags
- -xCORE-AVX512
  - Turns on optimization for the Skylake/Cascade Lake processor
- -qopt-zmm-usage=high
  - Improves performance for some code

STREAM – Compiler Flags



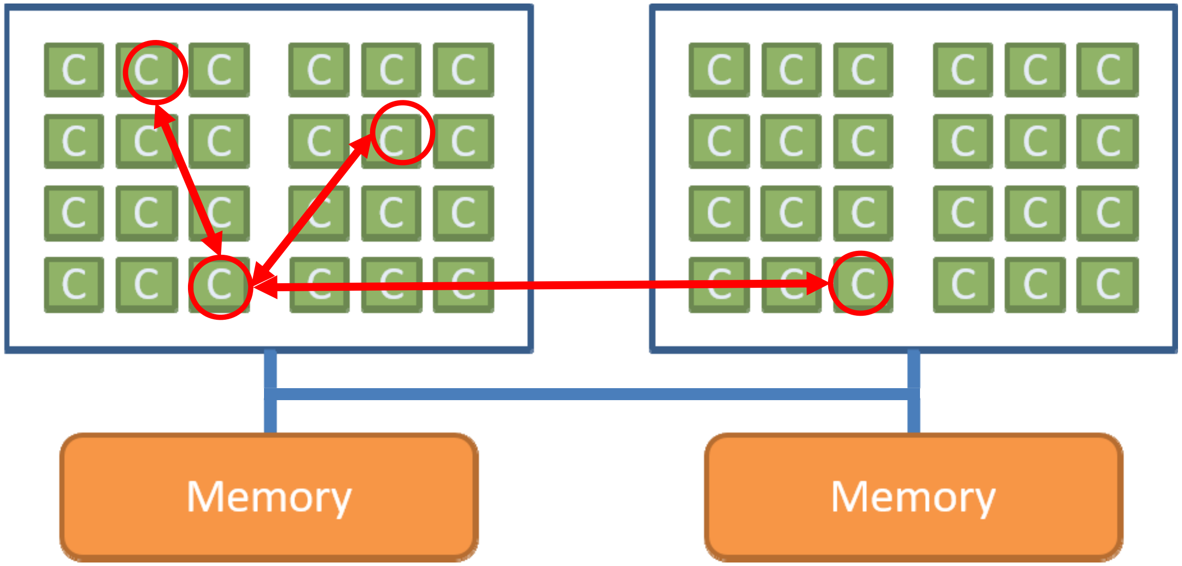
STREAM 5.10
Array size = 4000 MB
Intel 19.0.5
Default thread affinity

Thread Affinity



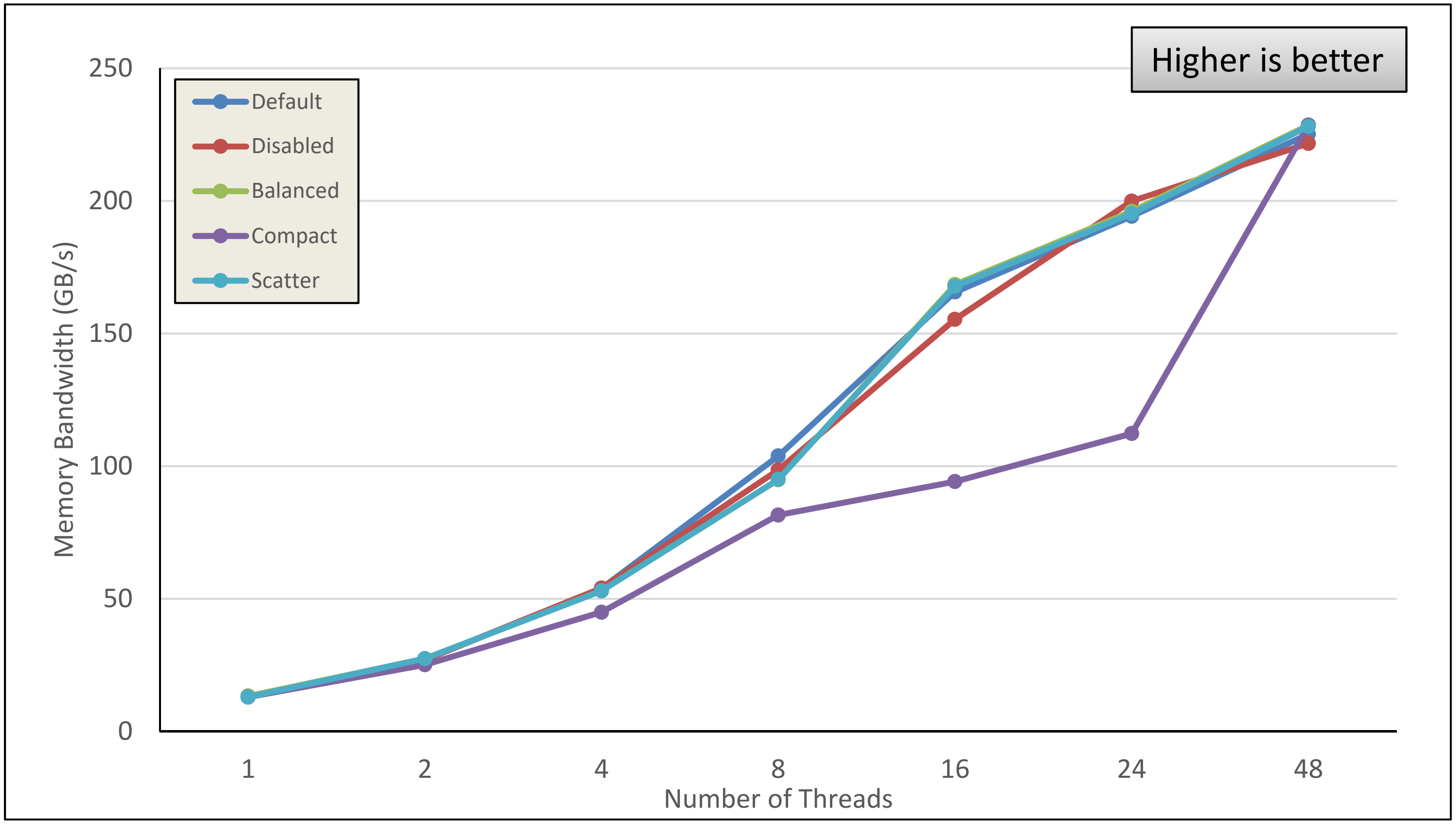
- The 48 cores on a node are grouped into 4 sets.
- The data exchange cost is not homogeneous.
- Depending on the data exchange pattern, how the threads are arranged could affect performance significantly.

Thread Affinity



- For Intel compiler, use the KMP_AFFINITY environment variable
- The options are “none”, “disabled”, “balanced”, “compact”, and “scatter”.

STREAM – Thread Affinity



STREAM 5.10
Array size = 4000 MB
Intel 19.0.5 with "-O3 –xCORE-AVX512 -qopt-zmm-usage=high"

HPL Benchmark

- High Performance Linpack
  - Standard benchmark for **CPU-bound** HPC applications
- Results
  - QB3: 2540 GFLOPS per node
  - QB2: 424 GFLOPS per node
  - Speedup = 6 (compared to 8.2 theoretical)

HPL 2.3 (Intel MKL version)
Problem size = 90% installed memory
MVAPICH2 2.3.3
Intel 19.0.5 with "-O3 –xCORE-AVX512"

HPCG Benchmark

- High Performance Conjugate Gradient
  - Standard benchmark for **memory-bound** HPC applications
- Results
  - QB3: 32.4 GFLOPS per node
  - QB2: 14.5 GFLOPS per node
  - Speedup: 2.2 (compared to 8.2 theoretical)

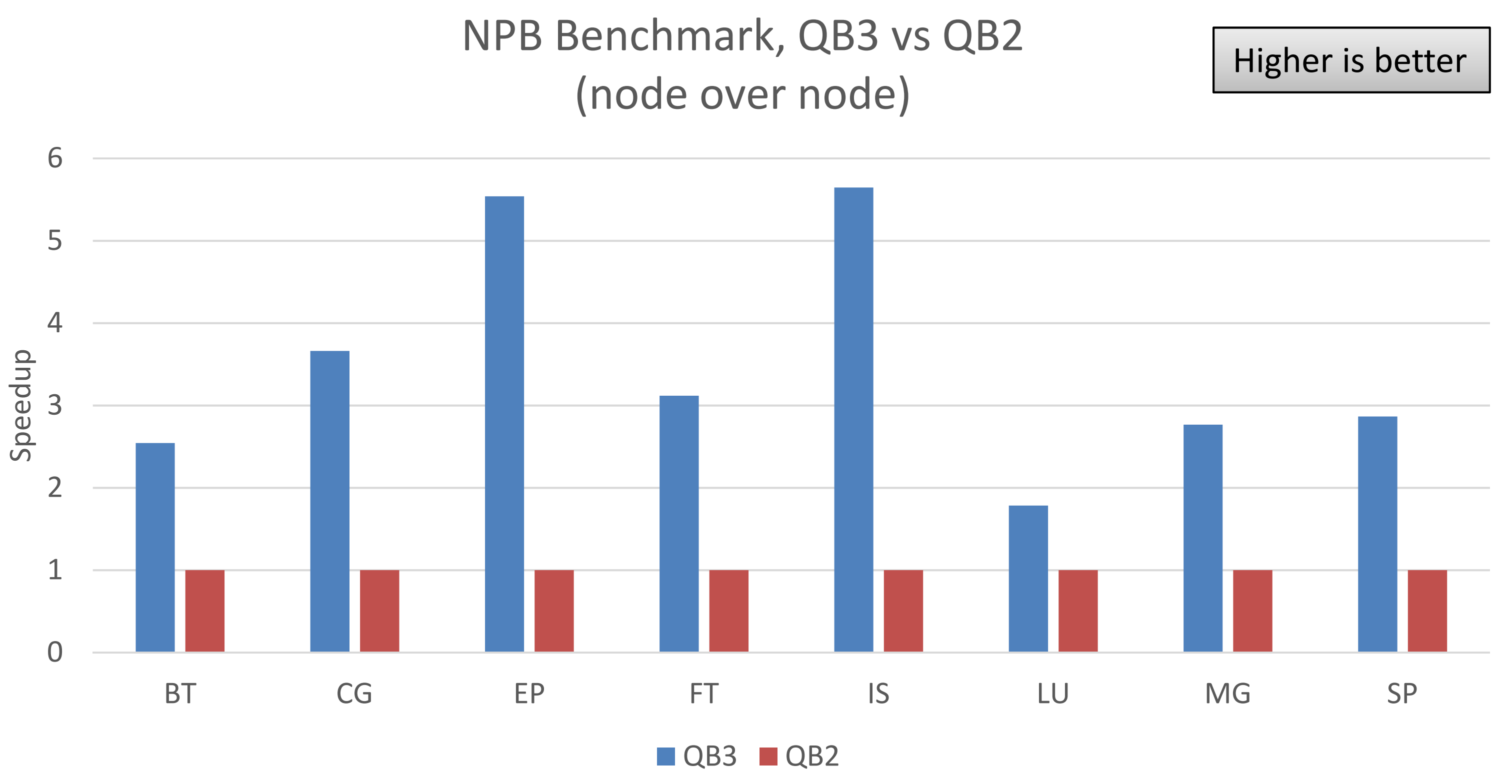
CENTER FOR COMPUTATION & TECHNOLOGY

HPCG 3.0 (Intel MKL version)
Problem size = 336 336 336
MVAPICH2 2.3.3
Intel 19.0.5 with "-O3 –xCORE-AVX512 -qopt-zmm-usage=high"

NPB Benchmark Suite

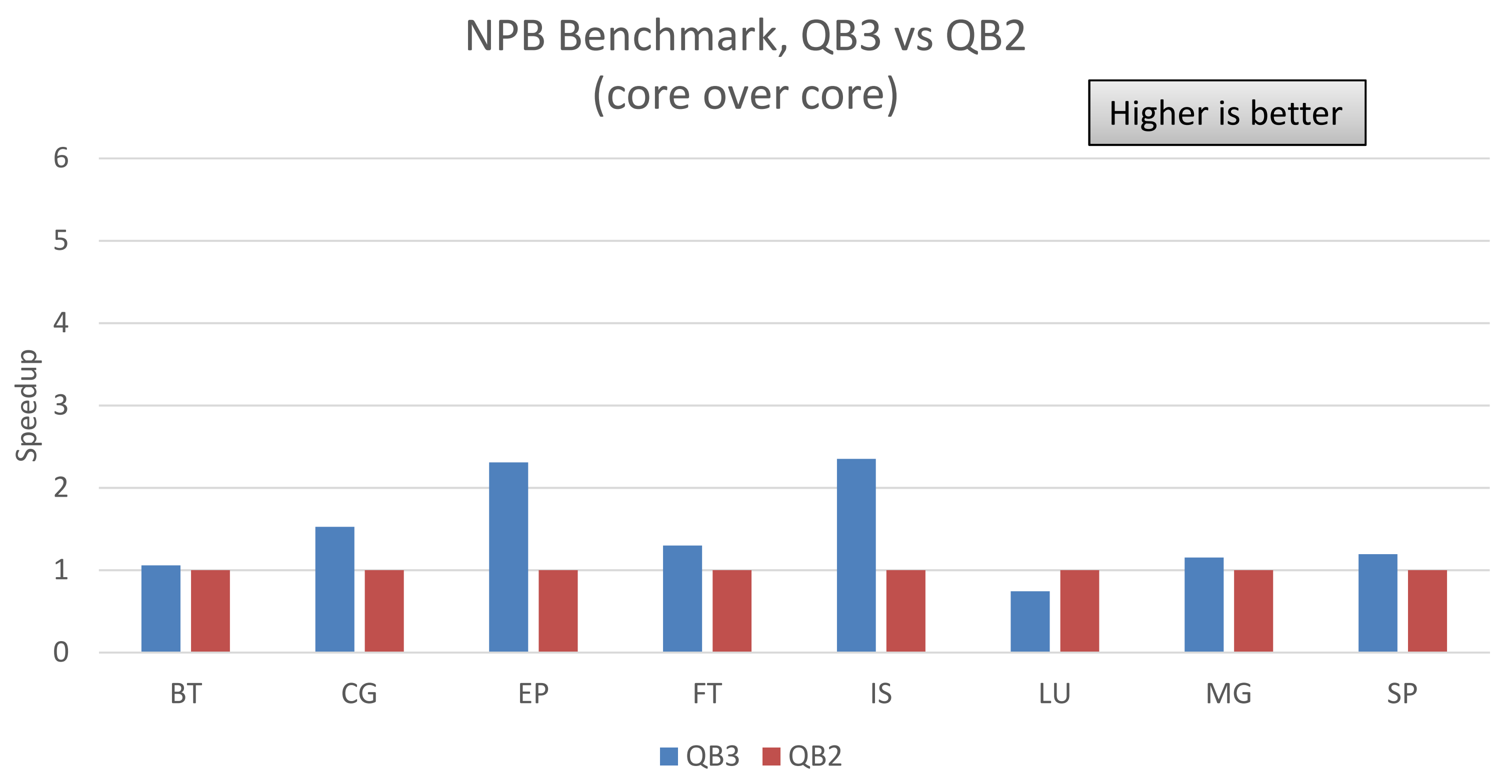
- NAS Parallel Benchmarks
  - a small set of programs derived from computational fluid dynamics applications
- Five kernels and three pseudo-applications
  - IS - Integer Sort, random memory access
  - EP - Embarrassingly Parallel
  - CG - Conjugate Gradient, irregular memory access and communication
  - MG - Multi-Grid on a sequence of meshes, long- and short-distance communication, memory intensive
  - FT - discrete 3D fast Fourier Transform, all-to-all communication
  - BT - Block Tri-diagonal solver
  - SP - Scalar Penta-diagonal solver
  - LU - Lower-Upper Gauss-Seidel solver

NPB Benchmarks – Node over Node



NPB 3.4.1, Class C
Intel 19.0.5 with "-O3 –xCORE-AVX512"
Default thread affinity

NPB Benchmarks – Core over Core

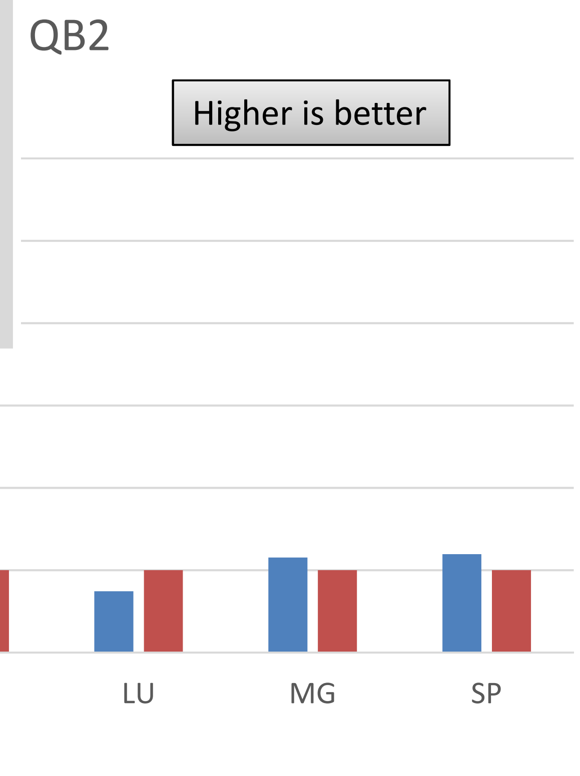


NPB 3.4.1, Class C
Intel 19.0.5 with "-O3 –xCORE-AVX512"
Default thread affinity

NPB Benchmarks – Core over Core

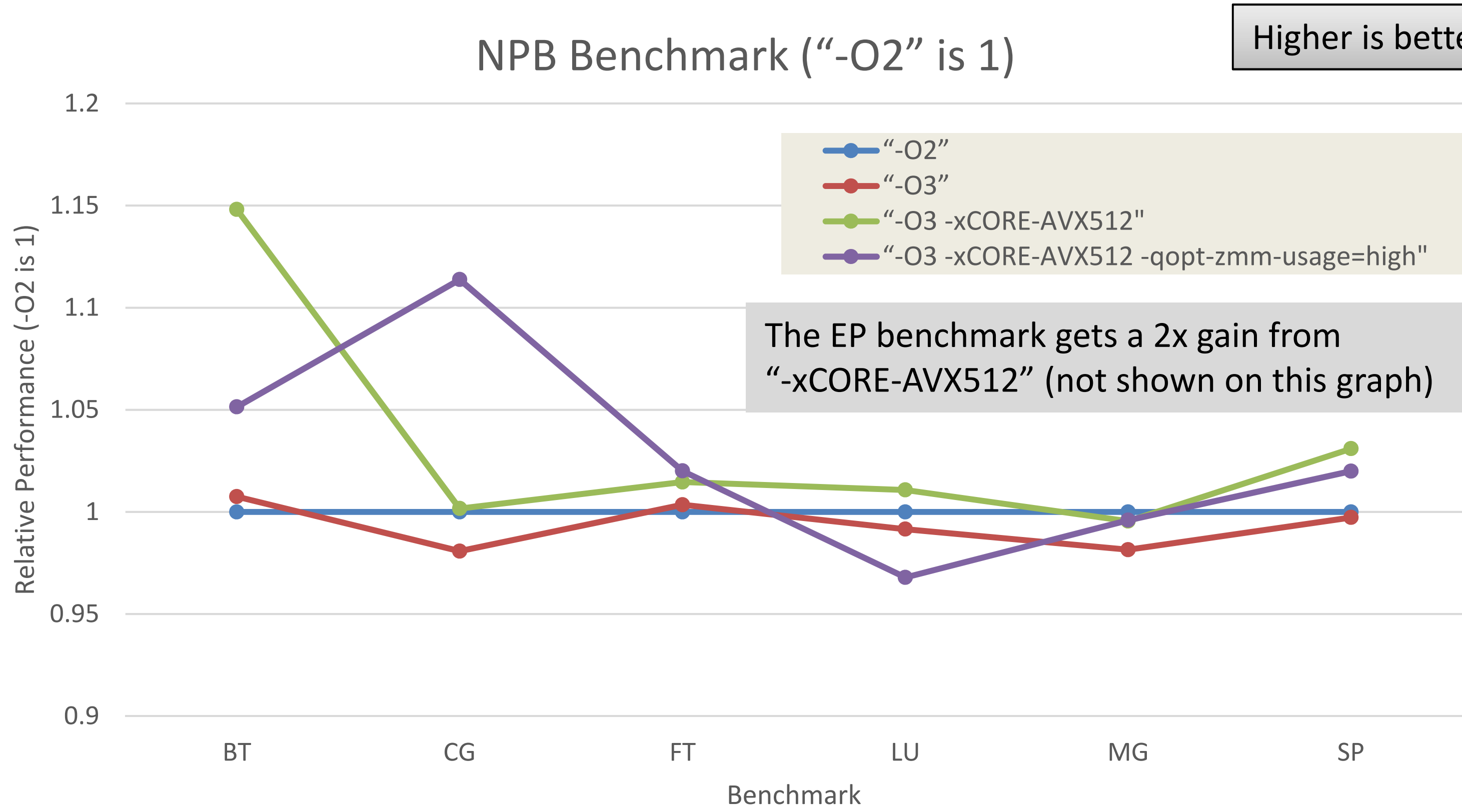
Takeway

- Performance gain on QB3 varies a lot from application to application.
- Core-over-core performance gain could be very limited.



NPB 3.4.1, Class C
Intel 19.0.5 with "-O3 –xCORE-AVX512"
Default thread affinity

NPB Results – Compiler Flags

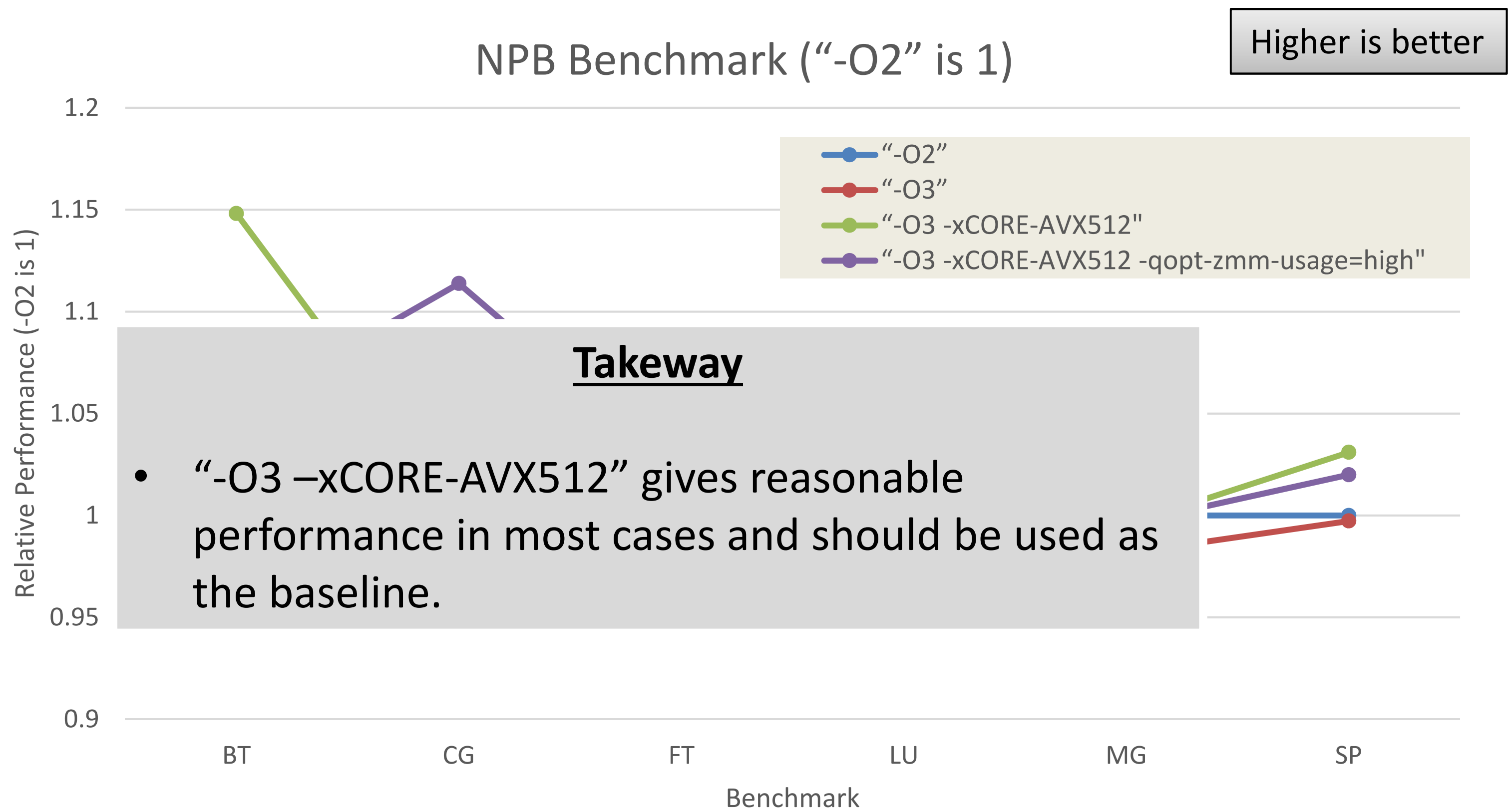


Higher is better

The EP benchmark gets a 2x gain from "-xCORE-AVX512" (not shown on this graph)

NPB 3.4.1, Class D
Intel 19.0.5
Default thread affinity

NPB Results – Compiler Flags

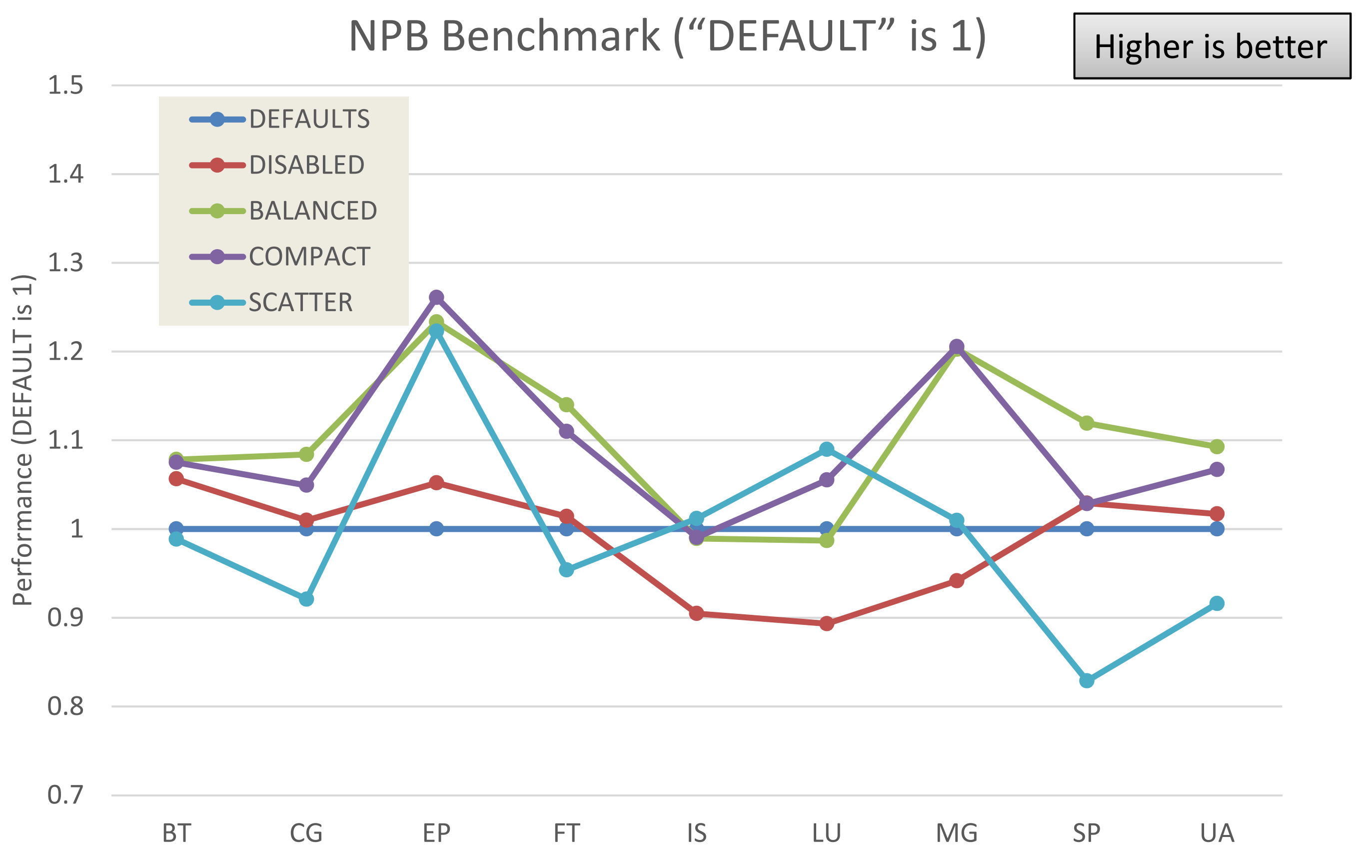


Takeway

- "-O3 –xCORE-AVX512" gives reasonable performance in most cases and should be used as the baseline.

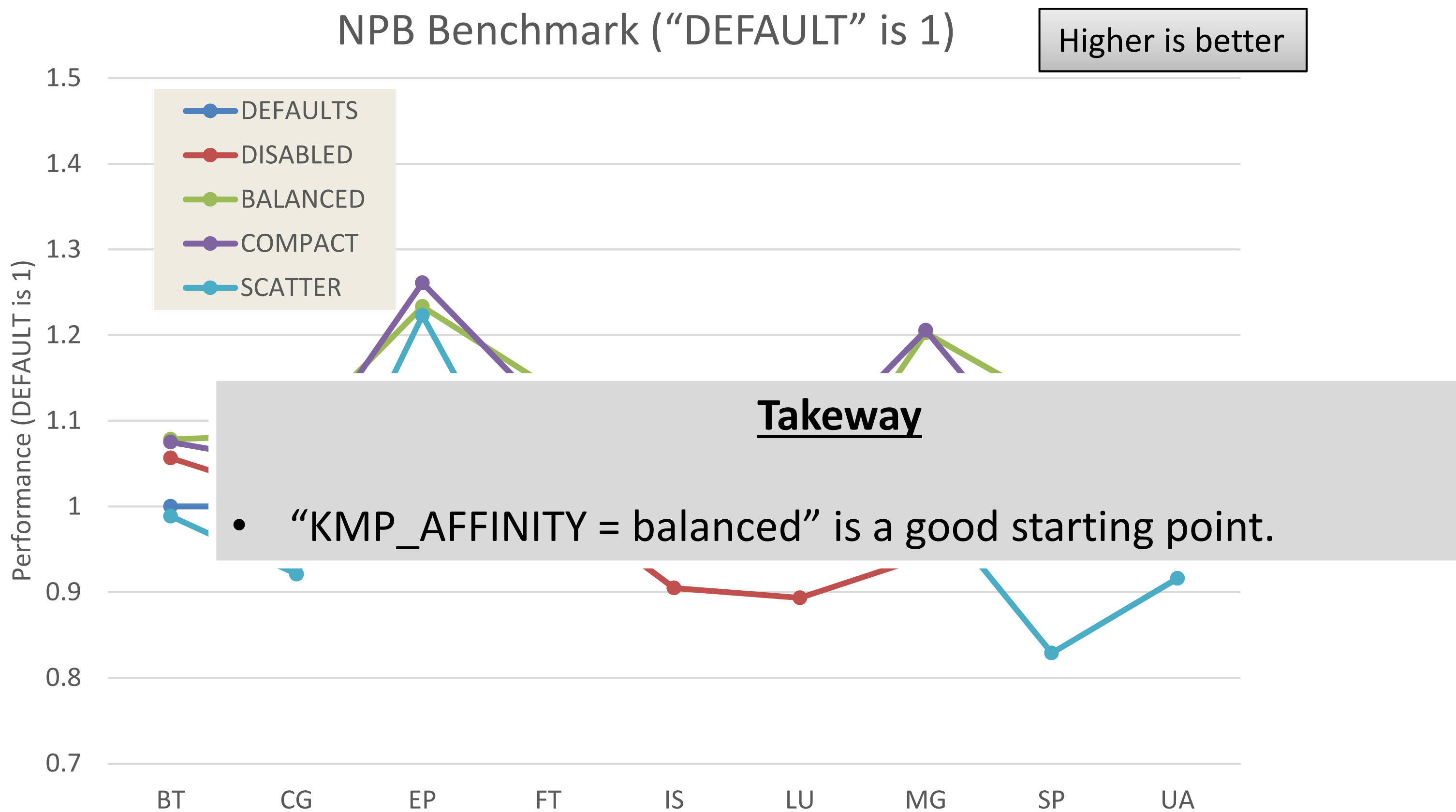
NPB 3.4.1, Class D
Intel 19.0.5
Default thread affinity

NPB Results – Thread Affinity



NPB 3.4.1, Class D
Intel 19.0.5 with “-O3 –xCORE-AVX512”

NPB Results – Thread Affinity



Takeway

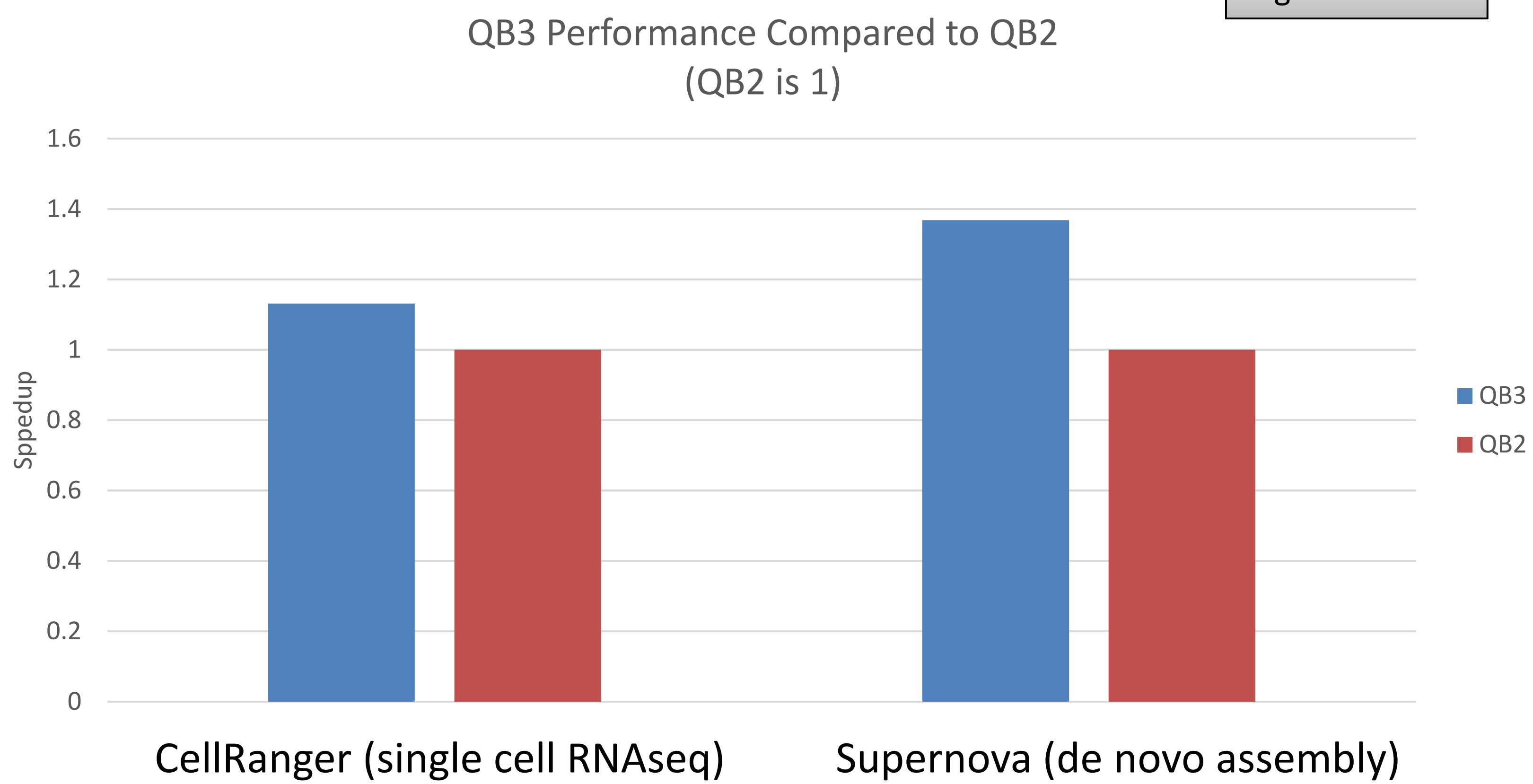
- "KMP_AFFINITY = balanced" is a good starting point.

NPB 3.4.1, Class D
Intel 19.0.5 with "-O3 –xCORE-AVX512"

CENTER FOR COMPUTATION & TECHNOLOGY

Bioinformatics

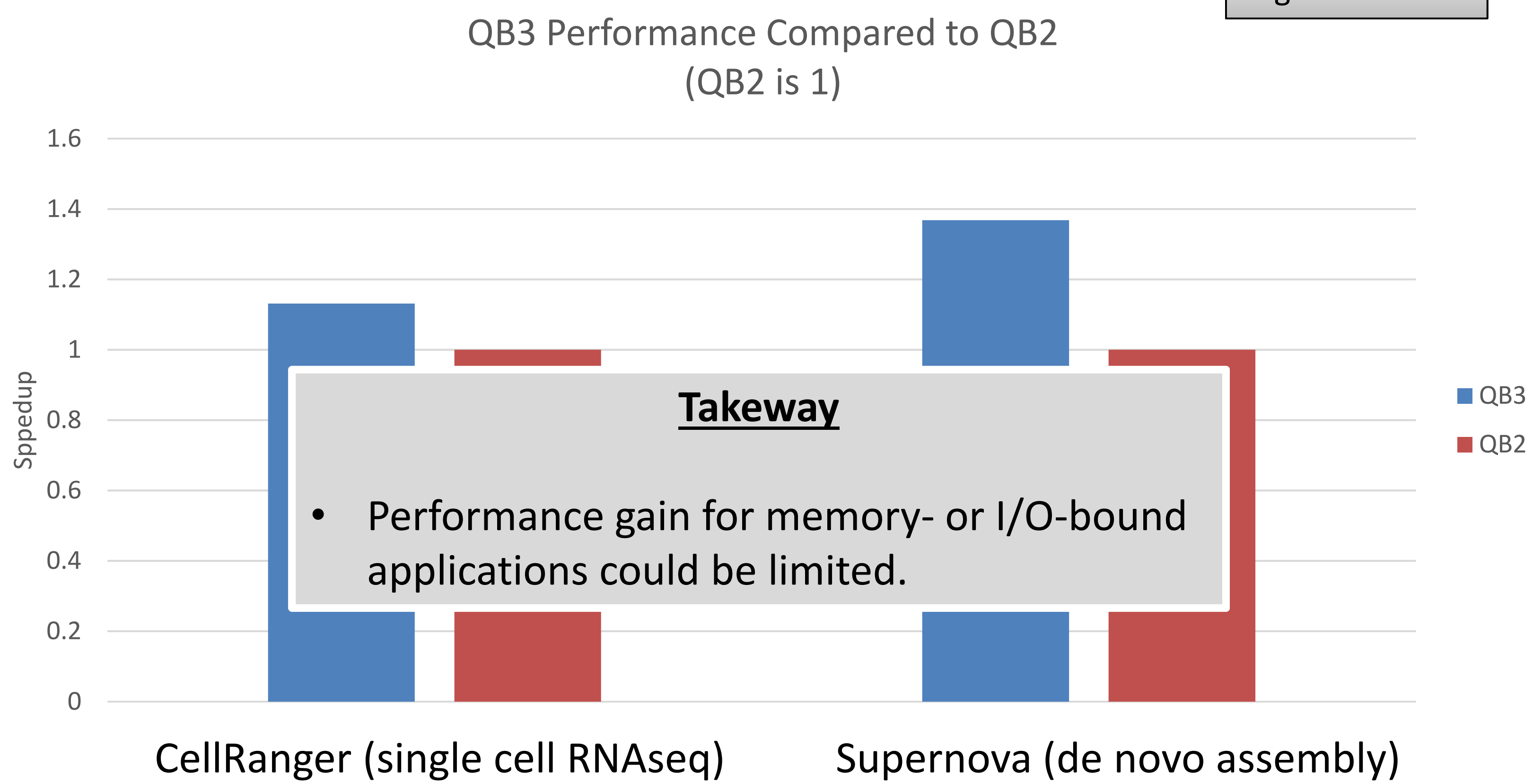
Higher is better



Both are binary distribution from 10x Genomics
Cellranger: mouse intestine cell, data size = 134 GB
Supernova: fruit fly genome, data size = 46 GB

Bioinformatics

Higher is better



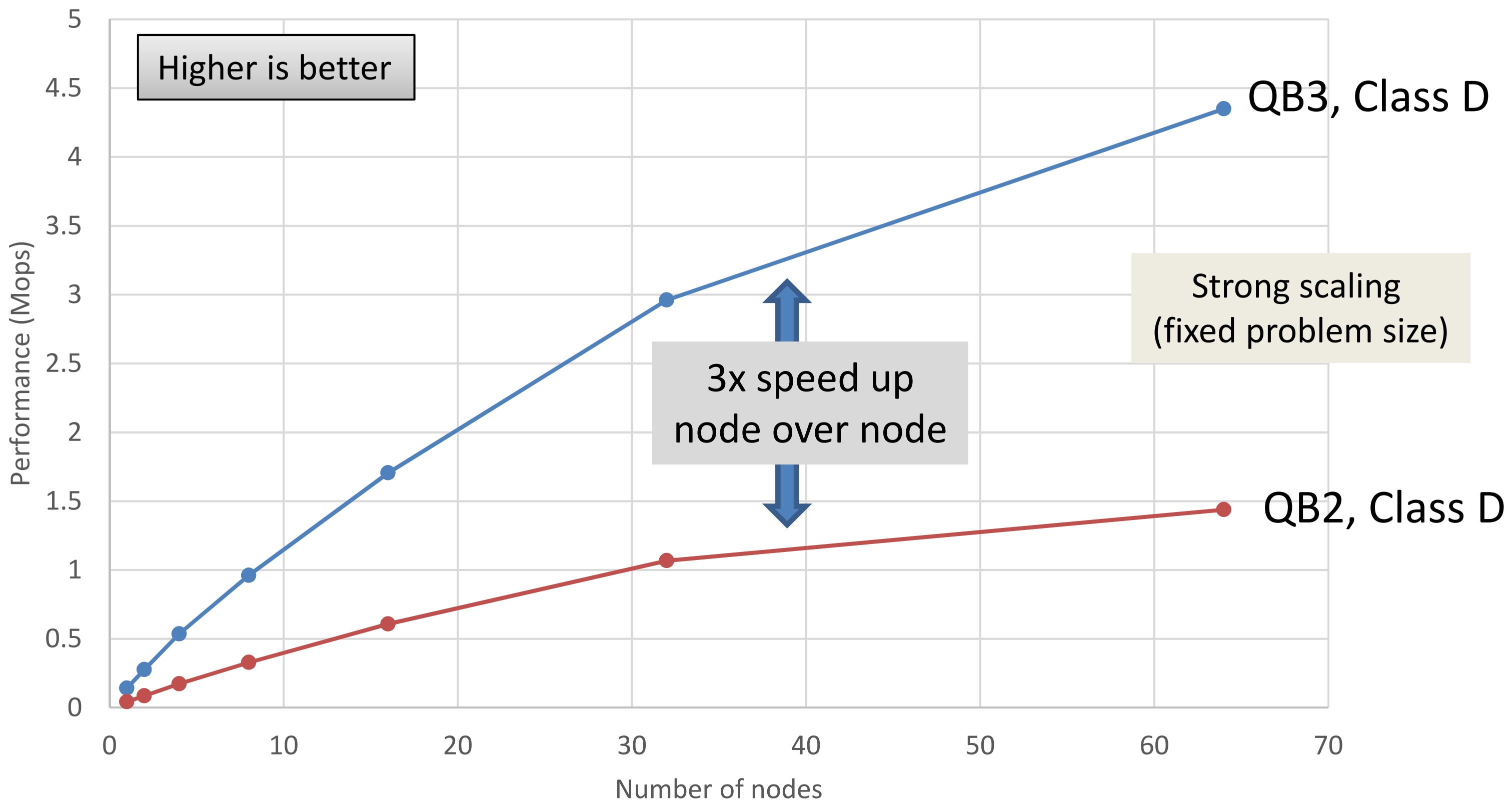
Takeway

- Performance gain for memory- or I/O-bound applications could be limited.

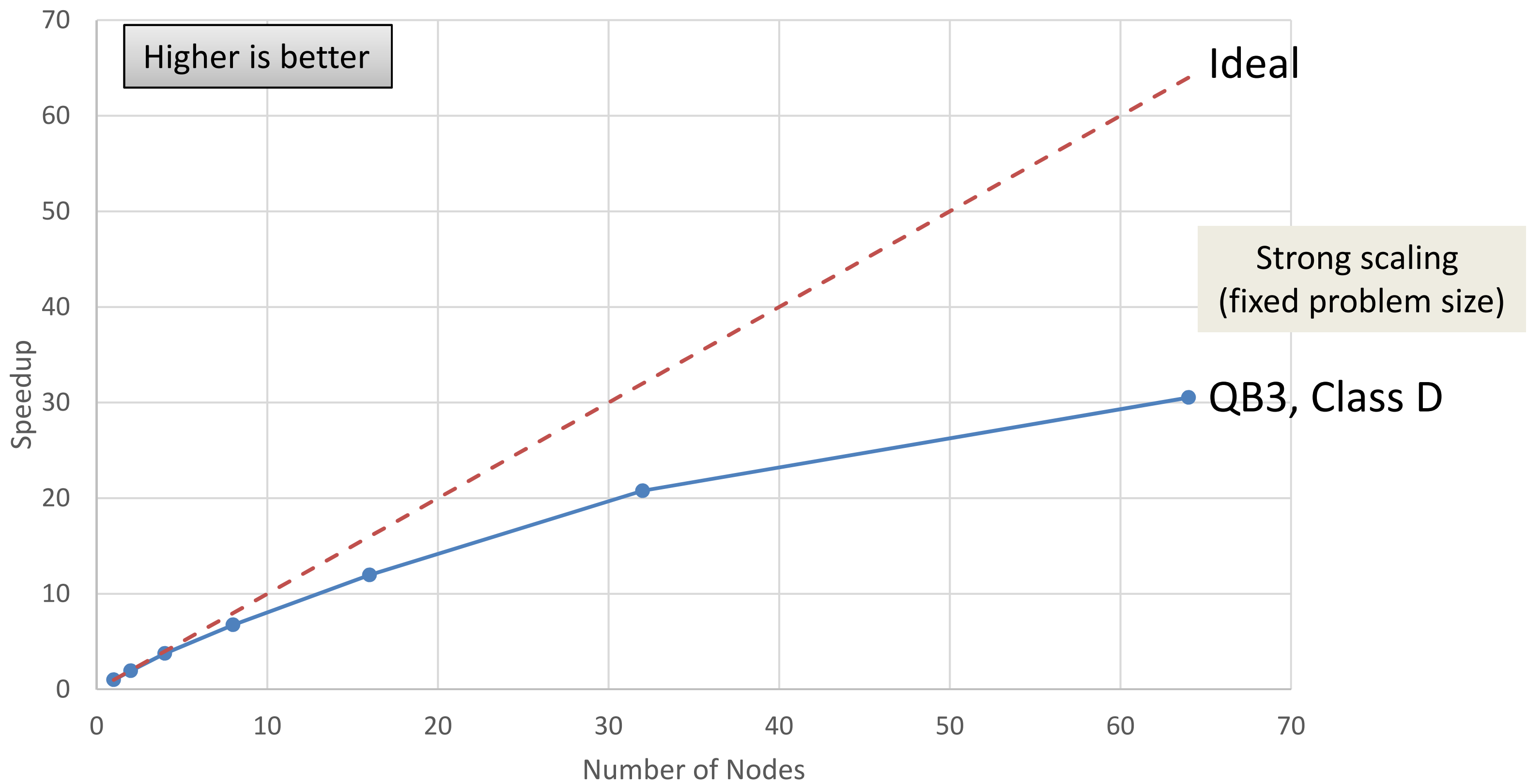
Both are binary distribution from 10x Genomics
Cellranger: mouse intestine cell, data size = 134 GB
Supernova: fruit fly genome, data size = 46 GB

Multi-node Performance

Pure MPI - NPB LU Benchmark

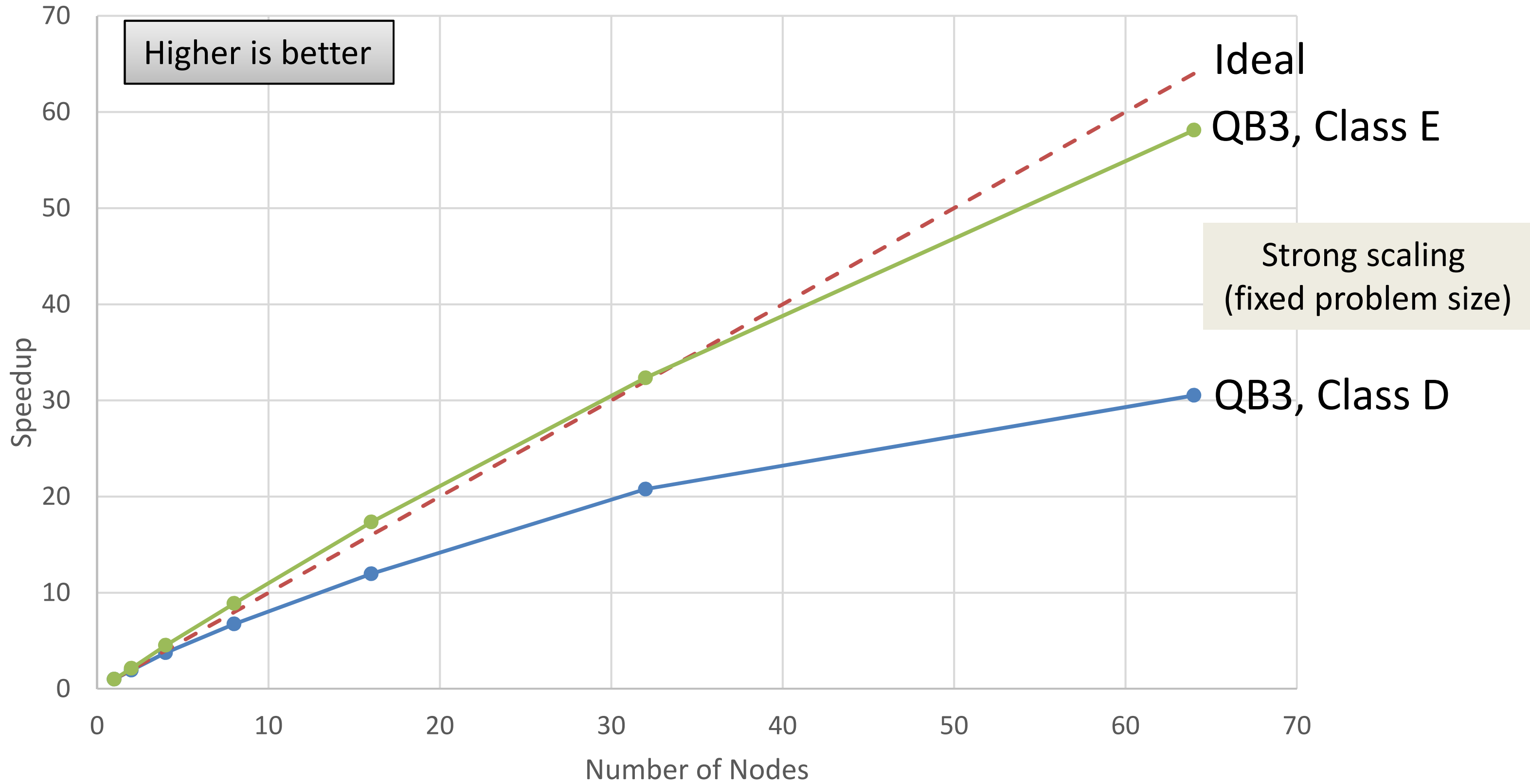


Pure MPI - NPB LU Benchmark



NPB 3.4.1, Class D and E
MVAPICH2 2.3.3
Intel 19.0.5 with "-O3 –xCORE-AVX512"

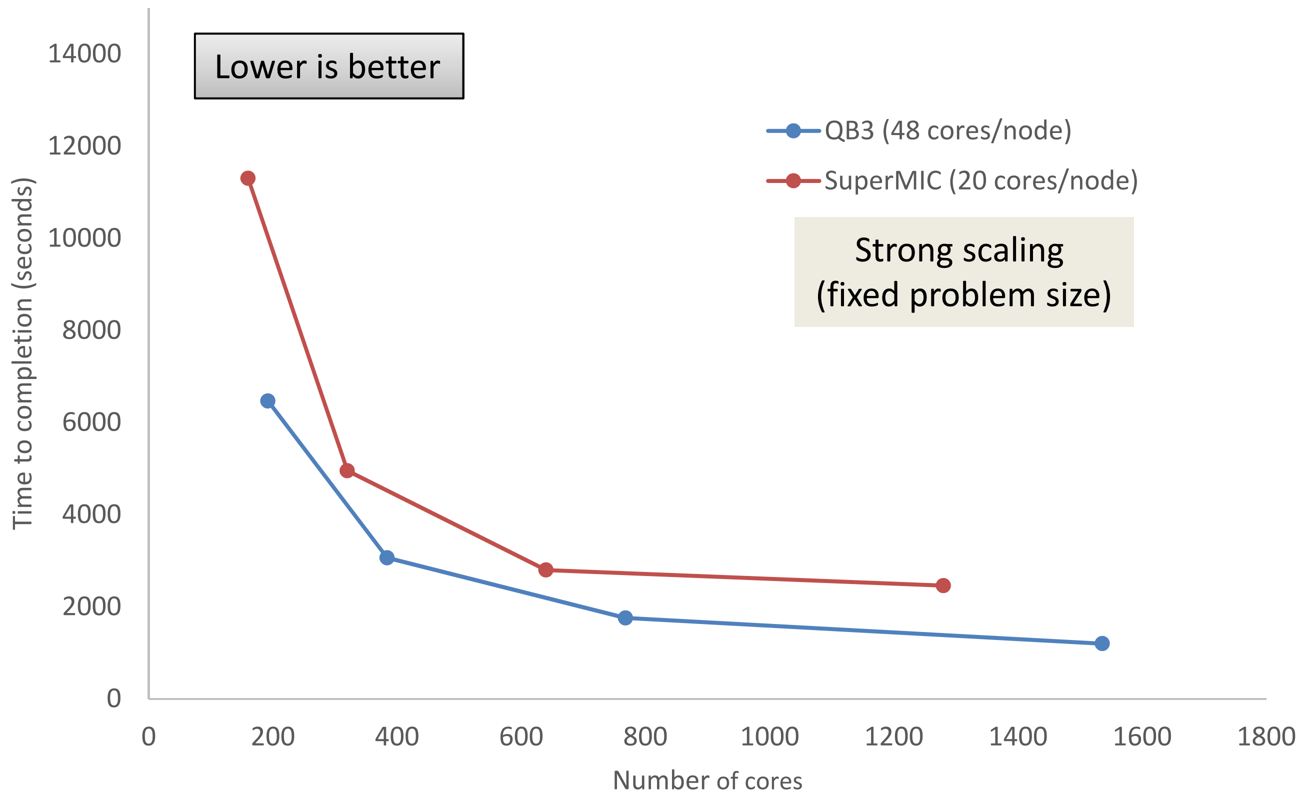
Pure MPI - NPB LU Benchmark



NPB 3.4.1, Class D and E
MVAPICH2 2.3.3
Intel 19.0.5 with "-O3 –xCORE-AVX512"

Pure MPI - PADCIRC

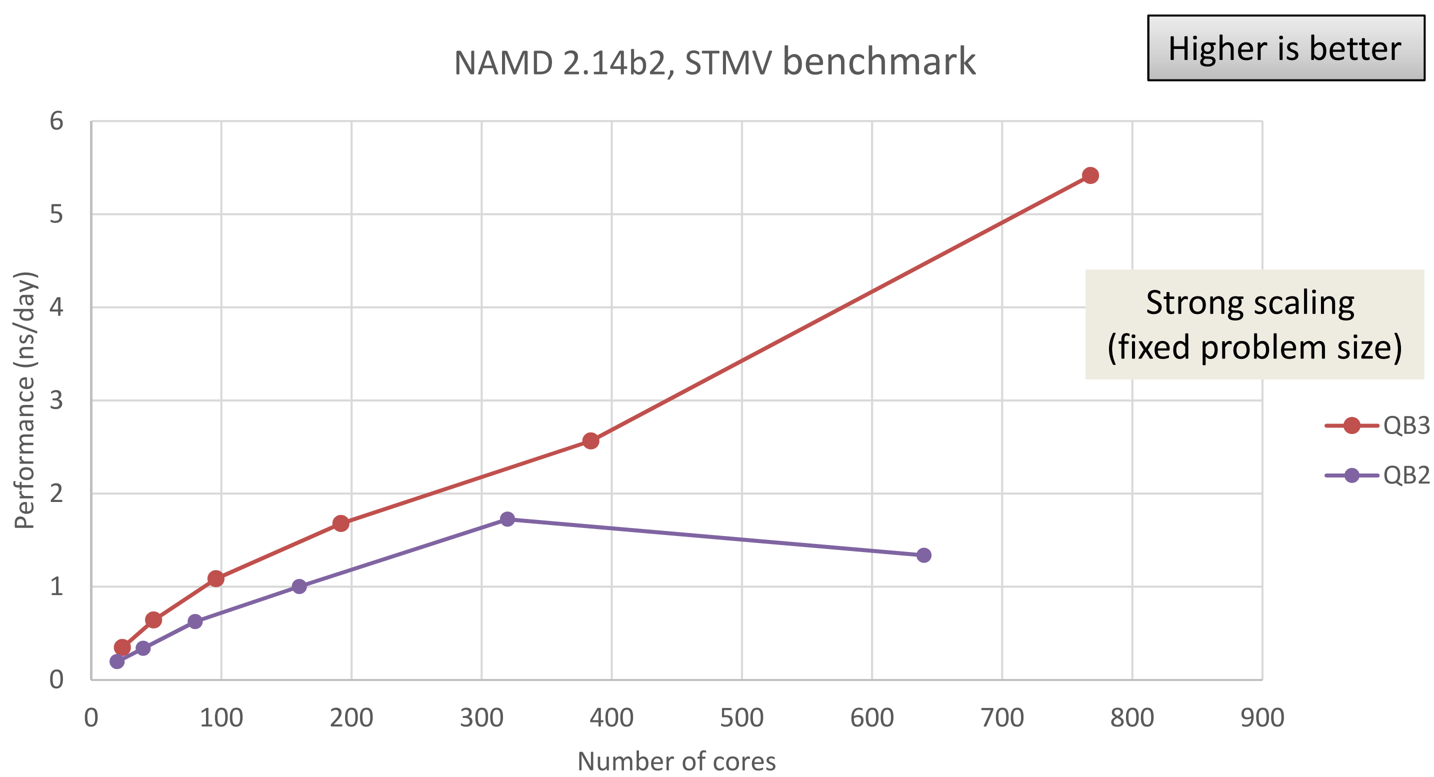
Storm surge modeling



PADCIRC version 55
MVAPICH2 2.3.3
Intel 19.0.5

NAMD

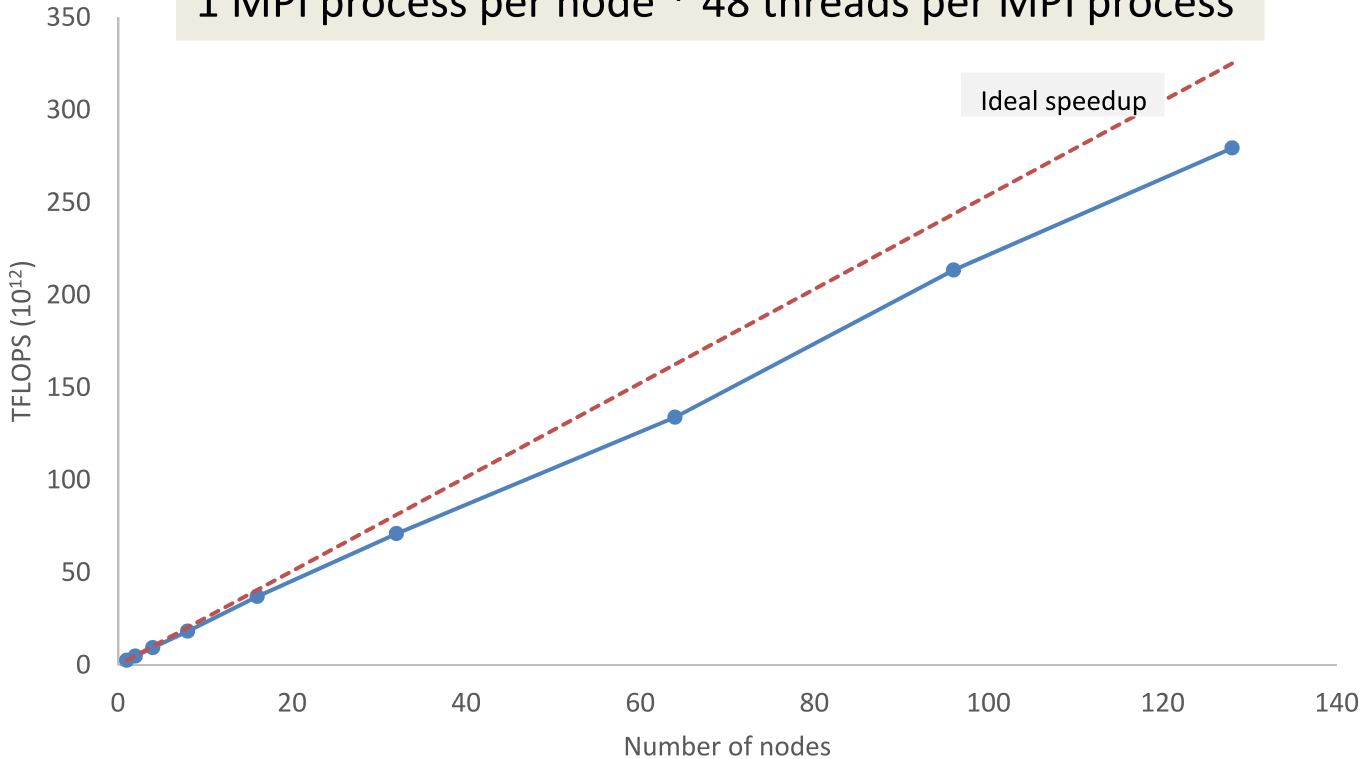
Molecular Dynamics



NAMD 2.14b2
Iverbs version of charmrun
Intel 19.0.5

Hybrid – HPL

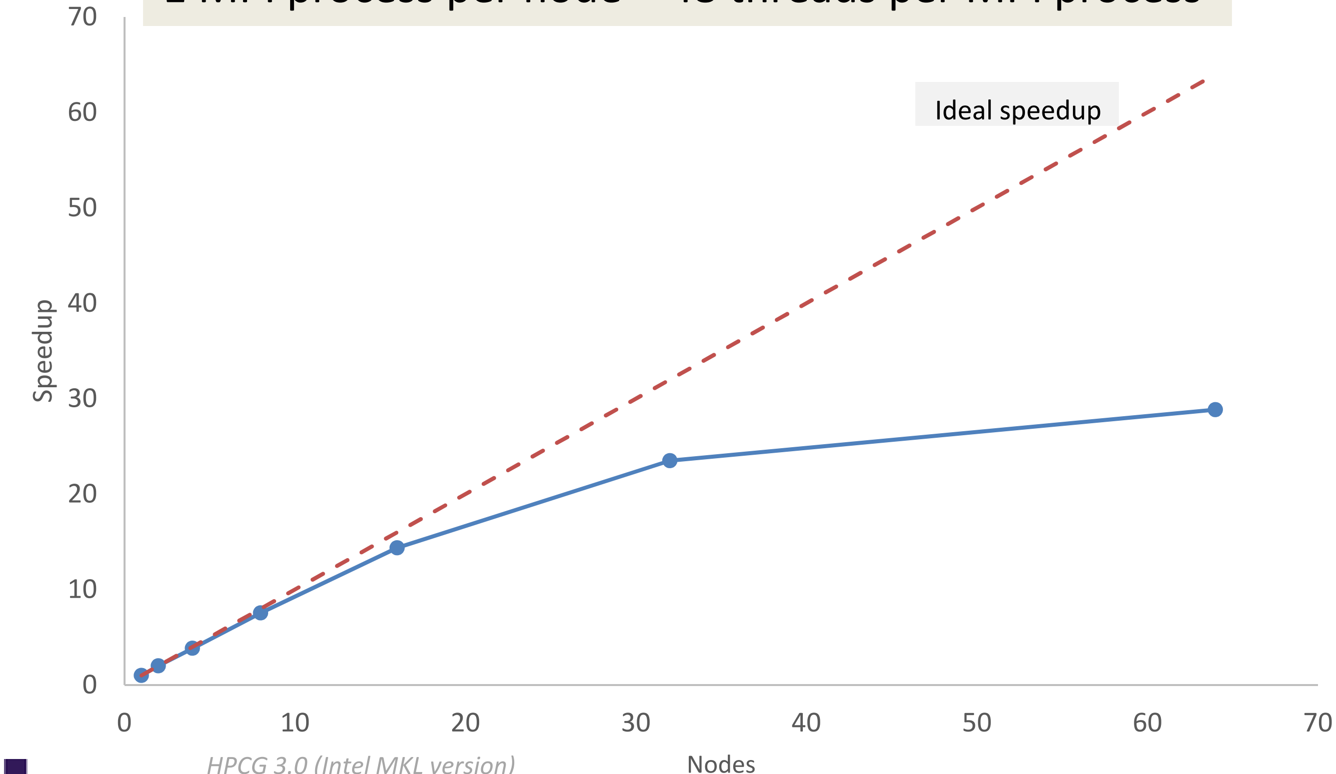
Weak scaling (problem size increases with core count)
1 MPI process per node * 48 threads per MPI process



HPL 2.3 (Intel MKL version)
Problem size = 90% installed memory
MVAPICH2 2.3.3
Intel 19.0.5 with "-O3 –xCORE-AVX512"

Hybrid - HPCG

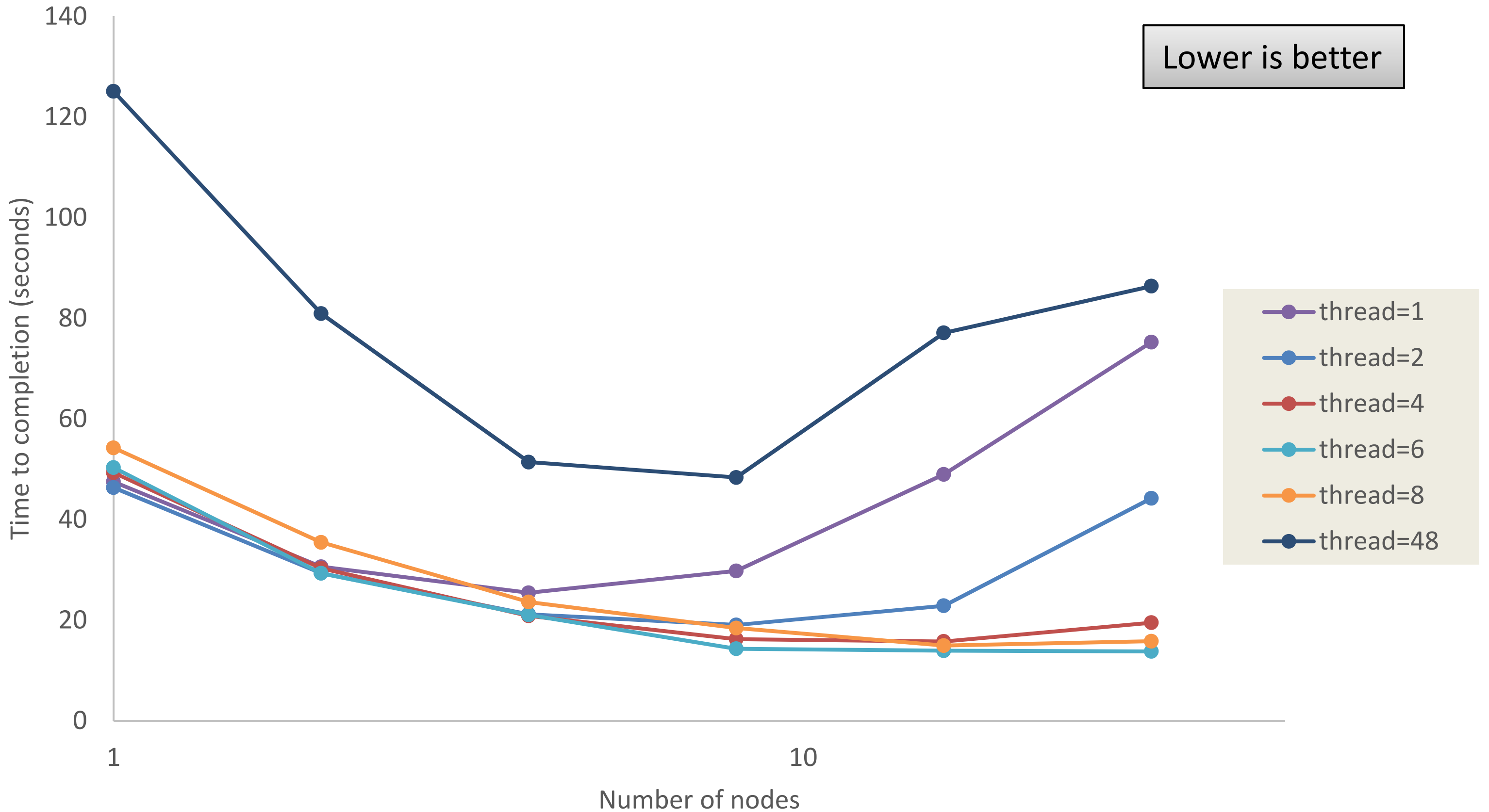
Weak scaling (problem size increases with core count)
1 MPI process per node * 48 threads per MPI process



HPCG 3.0 (Intel MKL version)
Problem size = 336 336 336
MVAPICH2 2.3.3
Intel 19.0.5 with "-O3 –xCORE-AVX512 -qopt-zmm-usage=high"

Hybrid - CP2K

Quantum Chemistry



CP2K 7.1
Intel MPI 2019u5
Intel 19.0.5 with "-O3 –xCORE-AVX512 -qopt-zmm-usage=high"

I/O Consideration

- You want to avoid letting your program accesses to disk excessively
  - Reading/writing hundreds of GBs to checkpoint or output files frequently
  - Running with thousands of MPI tasks all reading/writing individual files
- What you should and should not do
  - Avoid writing intermediate/checkpoint files unless necessary
  - Look for and use options that allow one or a few big files instead of files per process
  - Reduce the frequency of writing output files
  - Do not use /home for productive jobs – use /work instead

Takeaways

- In the majority cases, your application will run faster and scale better on QB3 (compared to QB2)
- That being said, how much faster depends on a lot of factors
- You need to figure that out before making a (educated) decision whether or not switch to QB3
- You need to run your own experiments

Takeways

- Baseline
  - Use "-O3 –xCORE-AVX512" to compile
  - The default settings work reasonably well in most cases
- Serial programs
  - The performance gain will be limited
- OpenMP programs
  - Try different KMP_AFFINITY settings
- MPI programs
  - Remember the scaling behavior depends on the problem size
- Hybrid programs
  - It does not always help
  - Finding the optimal number of threads could be tricky and certainly varies from application to application

Takeaways

- Baseline
  - Use "-O3 –xCORE-AVX512" to compile
  - The default settings work reasonably well in most cases
- Serial programs
  - The performance gain will be limited
- OpenM
  - Try different KMP_AFFINITY settings
- MPI programs
  - Remember the scaling behavior depends on the problem size
- Hybrid programs
  - It does not always help
  - Finding the optimal number of threads could be tricky and certainly varies from application to application

If you need help, let us know!