Performance Benchmarks and Tuning on SM-3

Le Yan

LSU CENTER FOR COMPUTATION & TECHNOLOGY

Objectives

- Show application performance benchmarks on SuperMike-3
- 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

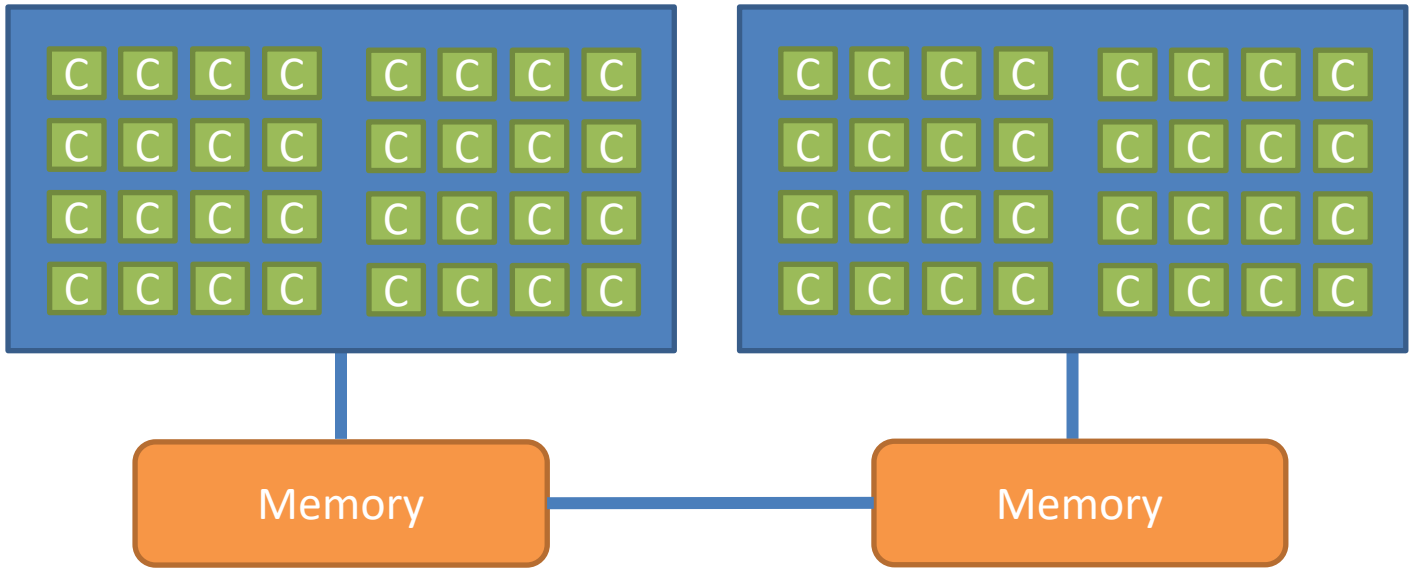
Outline

- Summary of SuperMike-3 architecture
- Single node performance
- Multi-node performance

Summary of SM-3 architecture

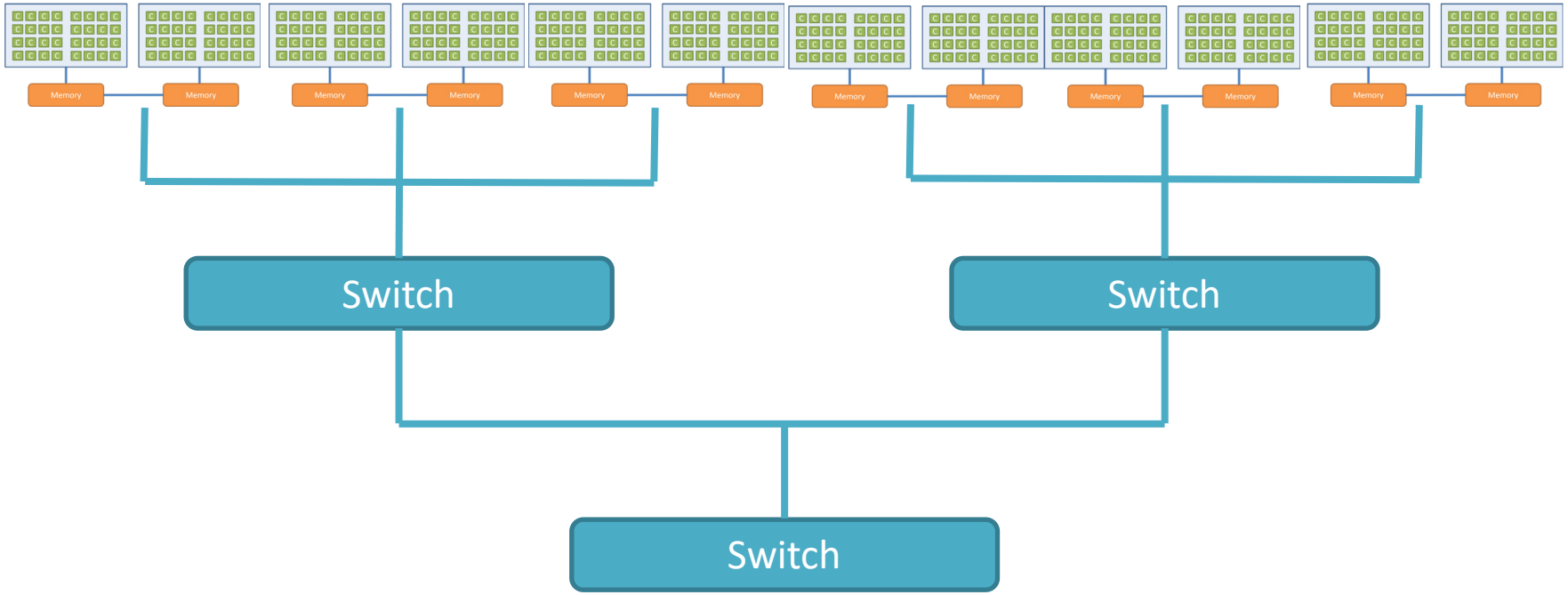
SM-3 Architecture

Node level view

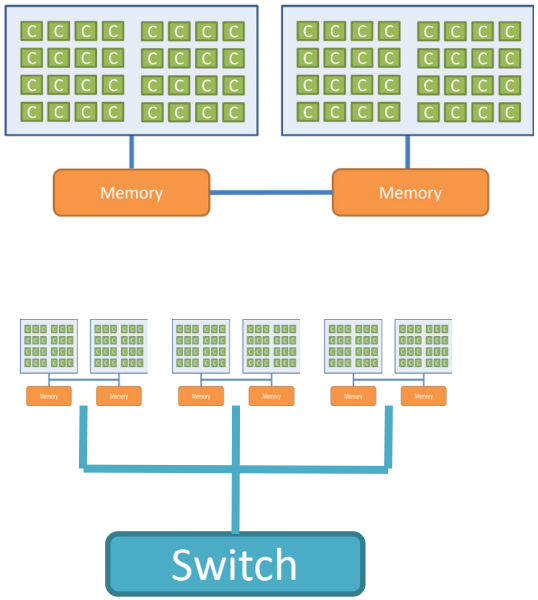


SM-3 Architecture

Cluster level view

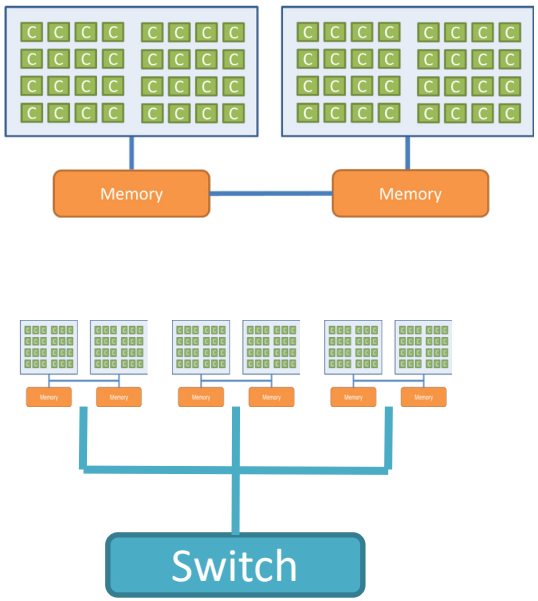


Parallel Paradigms



	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

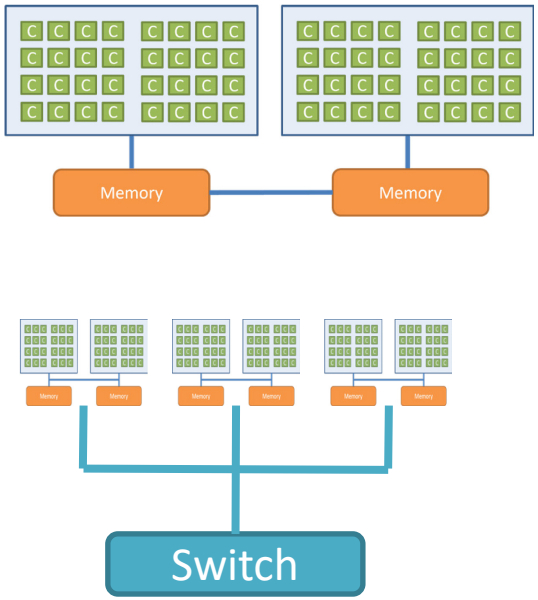


	Pro	Con
Intranode	- Low latency, high bandwidth - Implicit - Fine granularity - Dynamic load balancing	- Shared memory system - d to one
Internode	- Scalability beyond	- High latency, low munication - Hard to load balancing

OpenMP (Multi-thread)

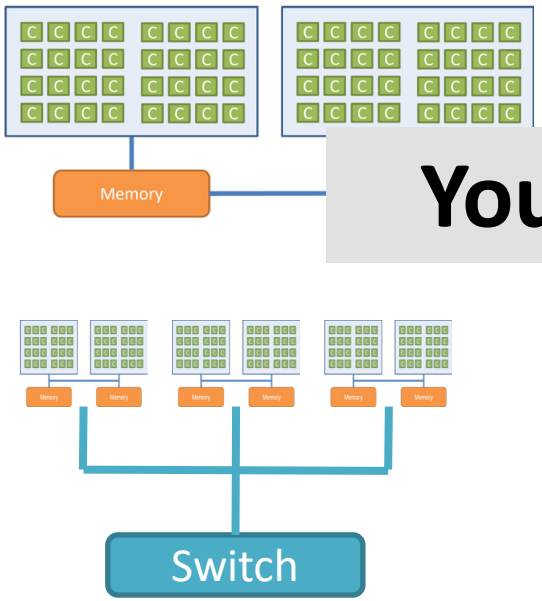
MPI (Multi-process)

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

Your mileage may vary!

(well-written) MPI programs might hurt the performance

- Hybrid helps to
  - Reduce memory footprint
  - Extend scalability

Single Node Performance

SM-3 Specification

CPU	Intel Ice Lake (Xeon Platinum 8358) (2 sockets *32 cores/socket)
CPU frequency	2.6 G Hz
Floating operation per clock cycle (double precision)	32
Memory	256 GB DDR4
Interconnect	Mellanox 200 Gbps Infiniband

SM-3 vs SuperMIC (Node over Node)

	SM-3	SuperMIC
CPU frequency	2.6 x 10^9	2.8 x 10^9
CPU cores	64	20
Operation per cycle	32	8
Memory bandwidth	~115 GB/s	~48 GB/s
Interconnect	200 Gbps	56 Gbps

Node (theoretical) peak performance

SM-3: 64 cores/node * 2.6 x 10^9 cycles/second * 32 flop/cycle = 5.32 x 10^{12} flops

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

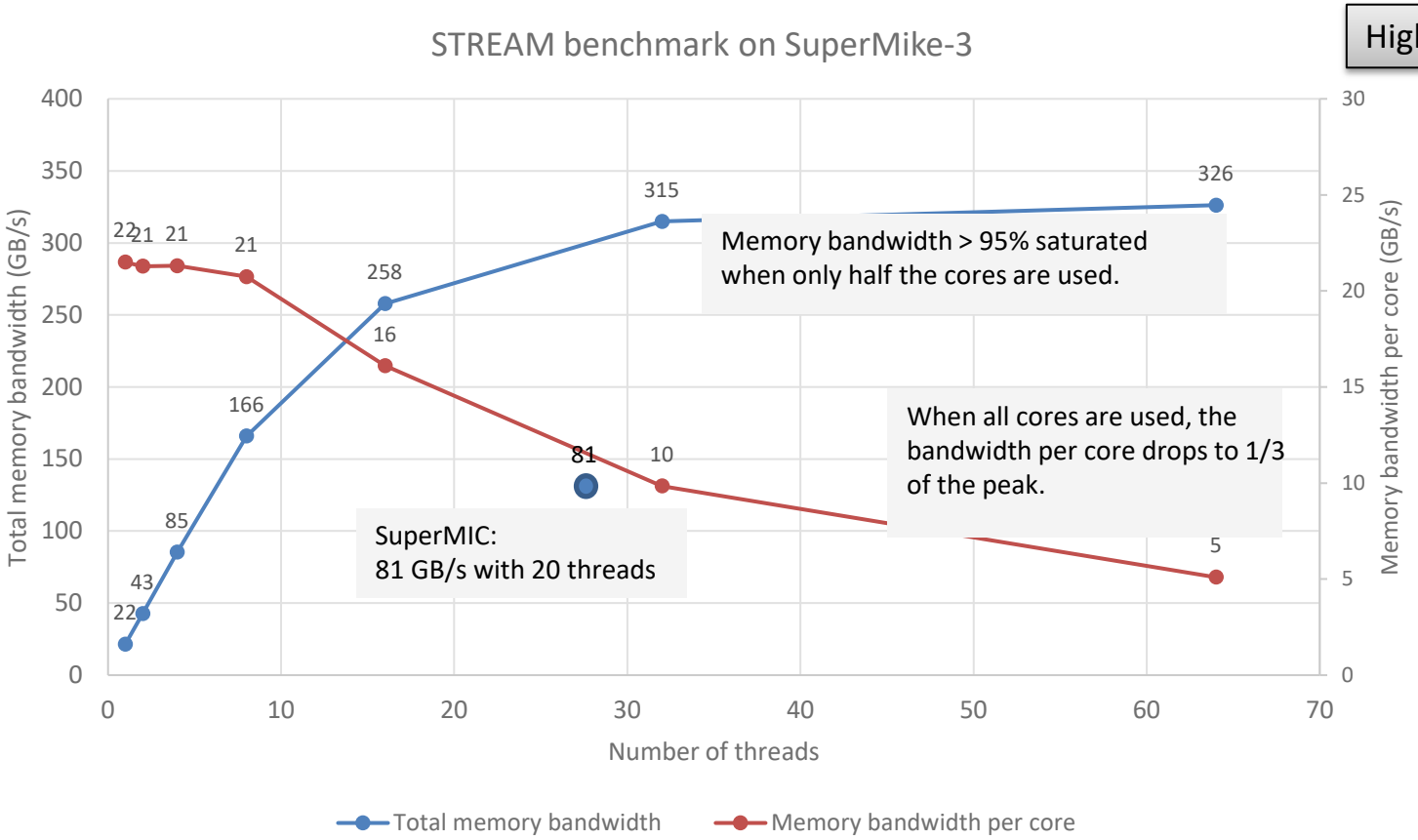
Theoretical speedup = 11.9

STREAM Benchmark

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

STREAM Benchmark - Results

Higher is better

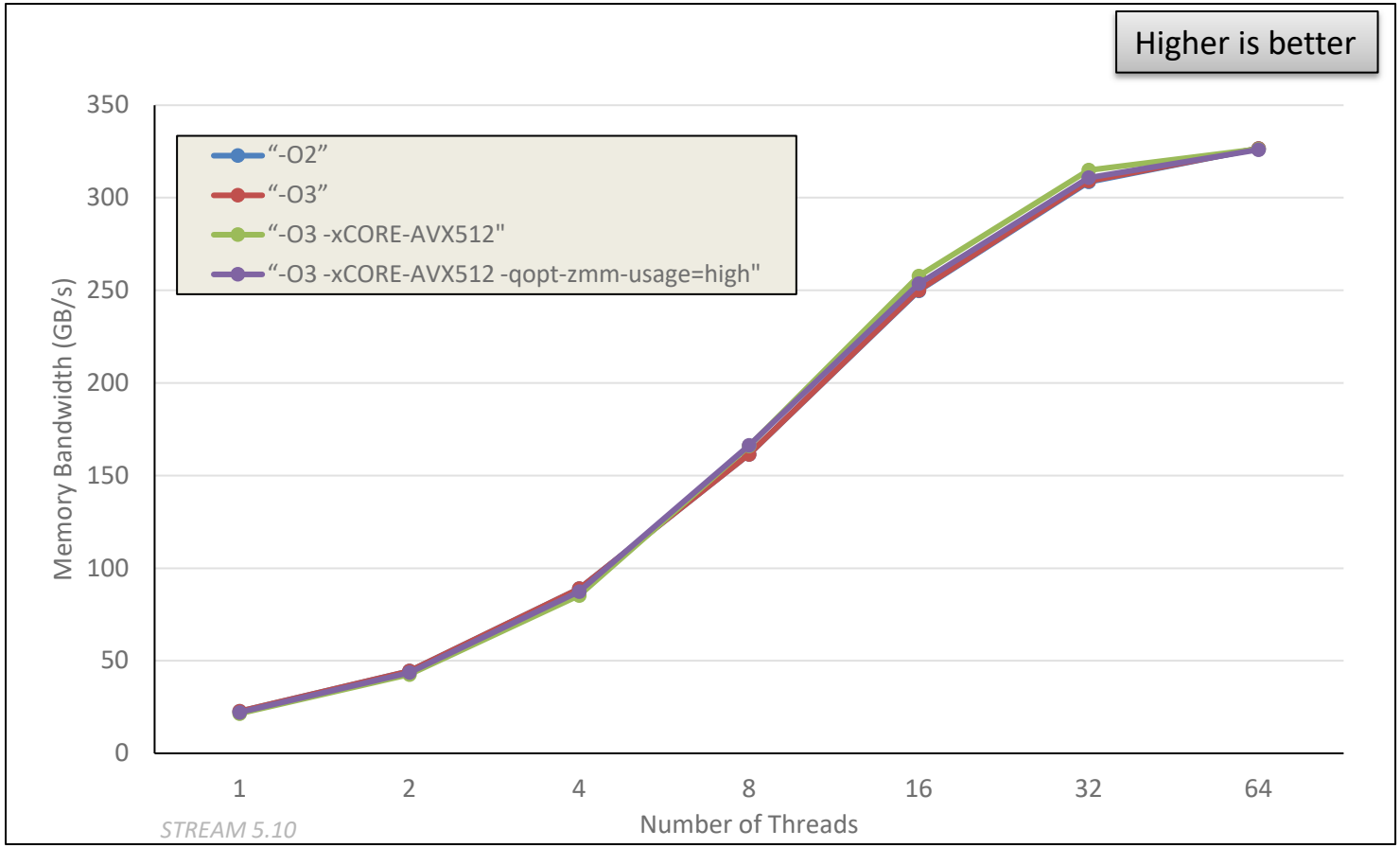


STREAM 5.10
Array size = 4000 MB
Intel 2021 .5.1 with "-O3 –xCORE-AVX512"
KMP_AFFINITY=balanced

Compiler Flags (Intel)

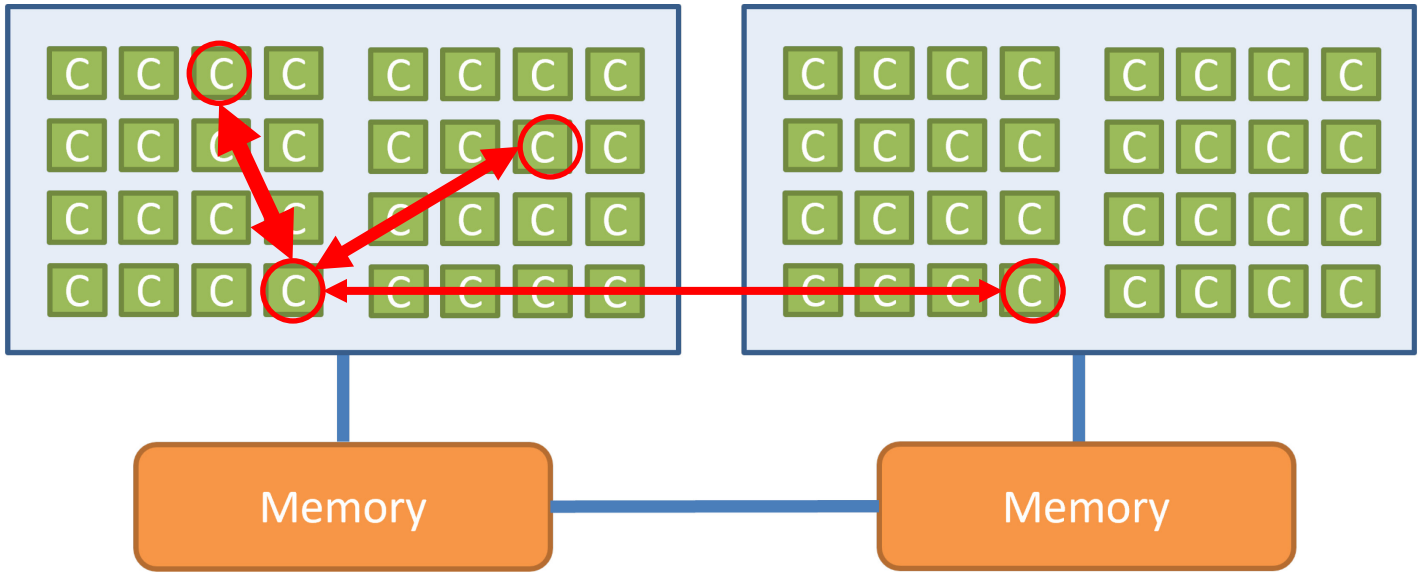
- -O2, -O3
  - Generic, aggregated optimization flags
- -xCORE-AVX512
  - Turns on optimization for the Skylake/Cascade Lake processor
- -qopt-zmm-usage=high
  - Improves performance for some code

STREAM – Compiler Flags



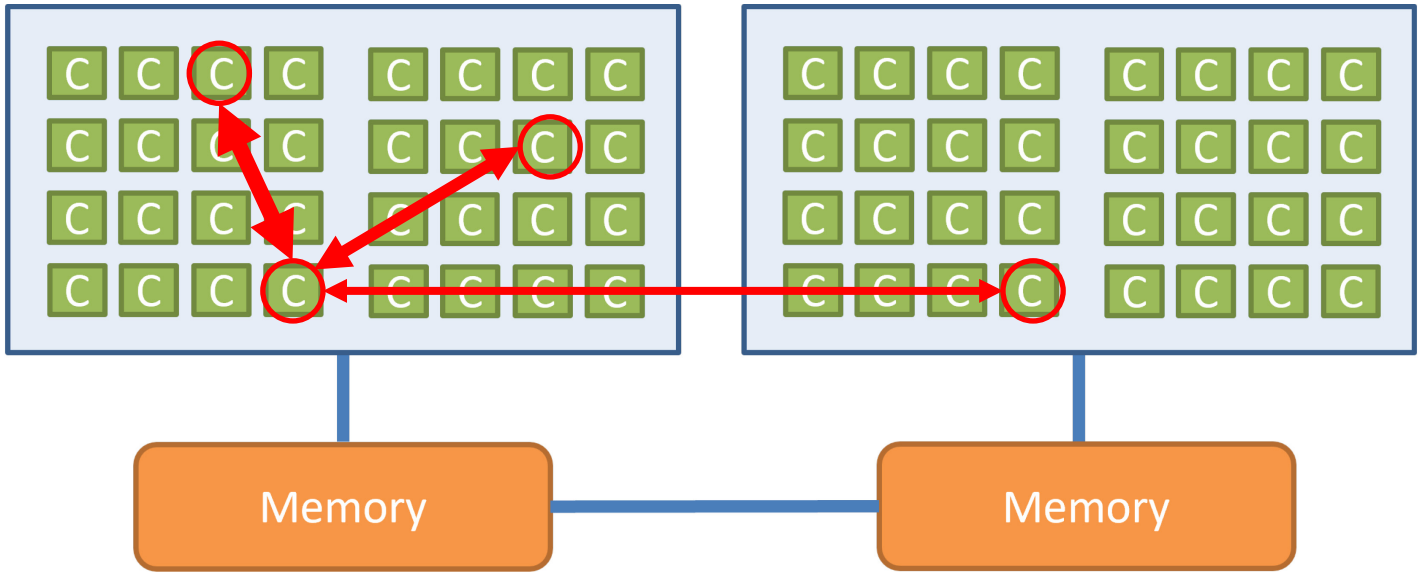
STREAM 5.10
Array size = 4000 MB
Intel 2021.5.1
KMP_AFFINITY=balanced

Thread Affinity



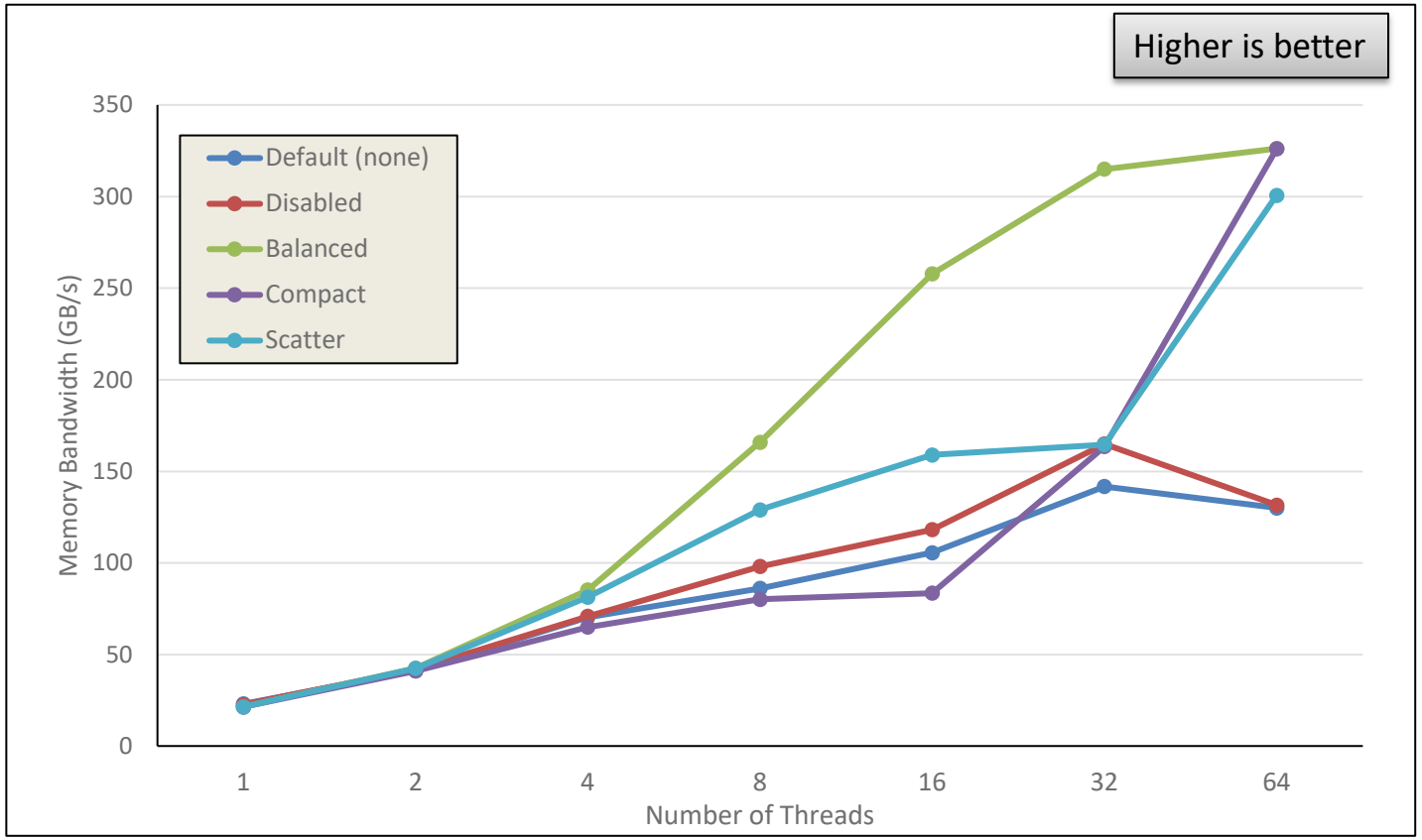
- The 64 cores on a SM-3 node are grouped into 4 sets.
- The data exchange cost is not homogeneous.
- Depending on the data exchange pattern, how the threads are arranged could affect performance significantly.

Thread Affinity



- Use the KMP_AFFINITY environment variable to control thread placement/affinity
- The options are “none” (default), “disabled”, “balanced”, “compact”, and “scatter”.

STREAM – Thread Affinity



STREAM 5.10
Array size = 4000 MB
Intel 2021.5.1 with "-O3 –xCORE-AVX512"

HPL Benchmark

- High Performance Linpack
  - Standard benchmark for **CPU-bound** HPC applications
- Results
  - SM-3: ~3800 GFLOPS per node
  - SuperMIC: 424 GFLOPS per node
  - Speedup = 9.0 (compared to 11.9 theoretical)

HPL 2.3 (Intel MKL version)
Problem size = 90% installed memory
Intel MPI 2021.5.0

HPCG Benchmark

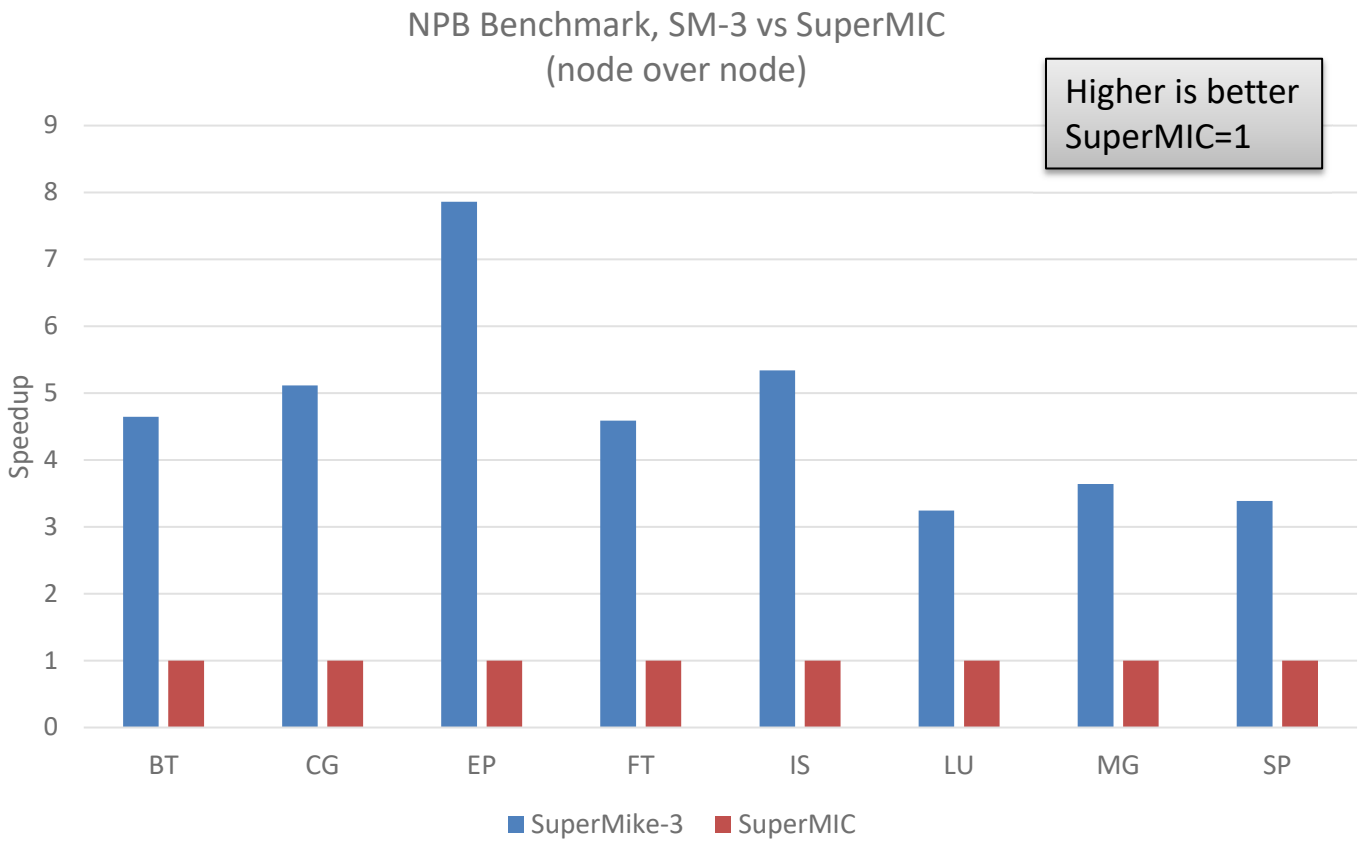
- High Performance Conjugate Gradient
  - Standard benchmark for **memory-bound** HPC applications
- Results
  - SM-3: 56.8 GFLOPS per node
  - SuperMIC: 14.5 GFLOPS per node
  - Speedup: 3.9 (compared to 11.9 theoretical)

HPCG 3.0 (Intel MKL version)
Problem size = 336 336 336
Intel MPI 2021.5.0

NPB Benchmark Suite

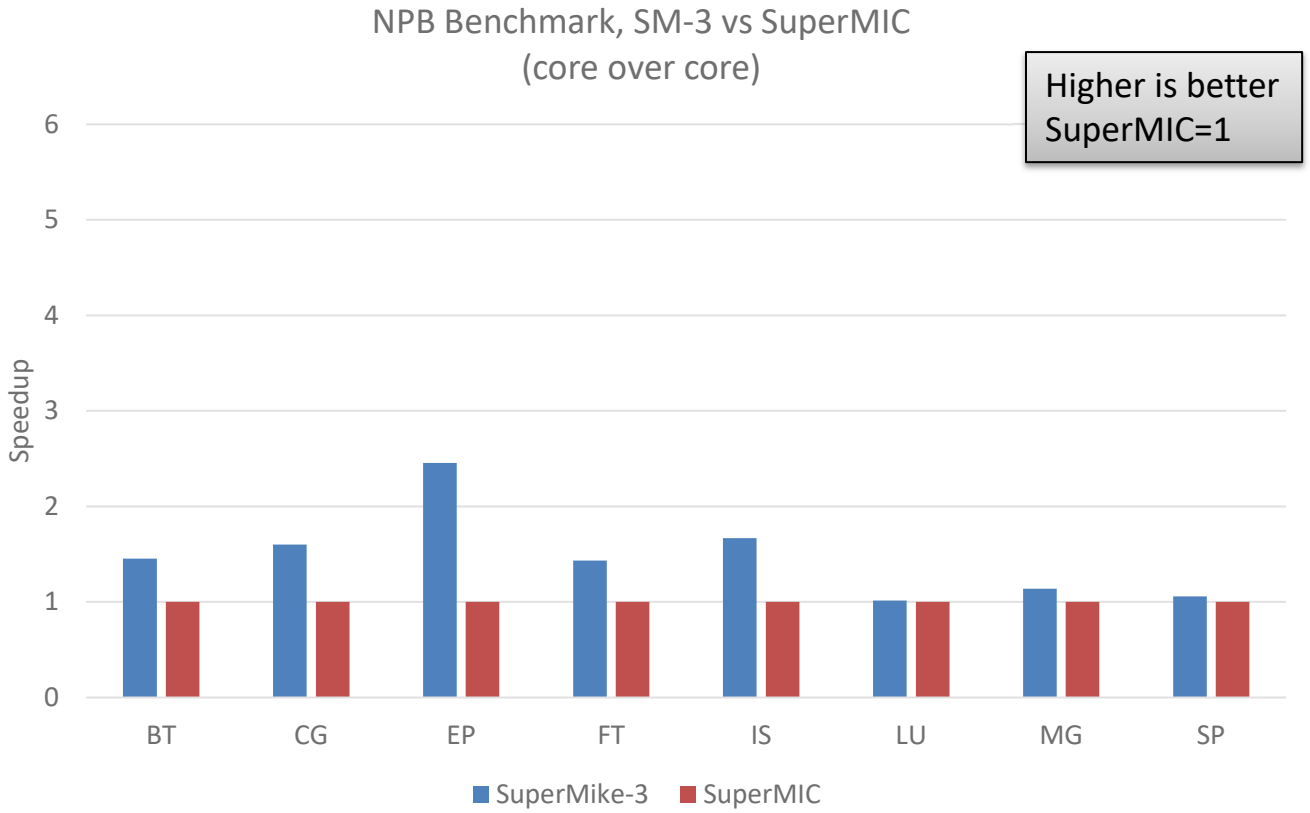
- NAS Parallel Benchmarks
  - a small set of programs derived from computational fluid dynamics applications
- Five kernels and three pseudo-applications
  - IS - Integer Sort, random memory access
  - EP - Embarrassingly Parallel
  - CG - Conjugate Gradient, irregular memory access and communication
  - 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 3.4.2, Class C
Intel 2021.5.1 with "-O3 –xCORE-AVX512"
KMP_AFFINITY=balanced

NPB Benchmarks – Core over Core

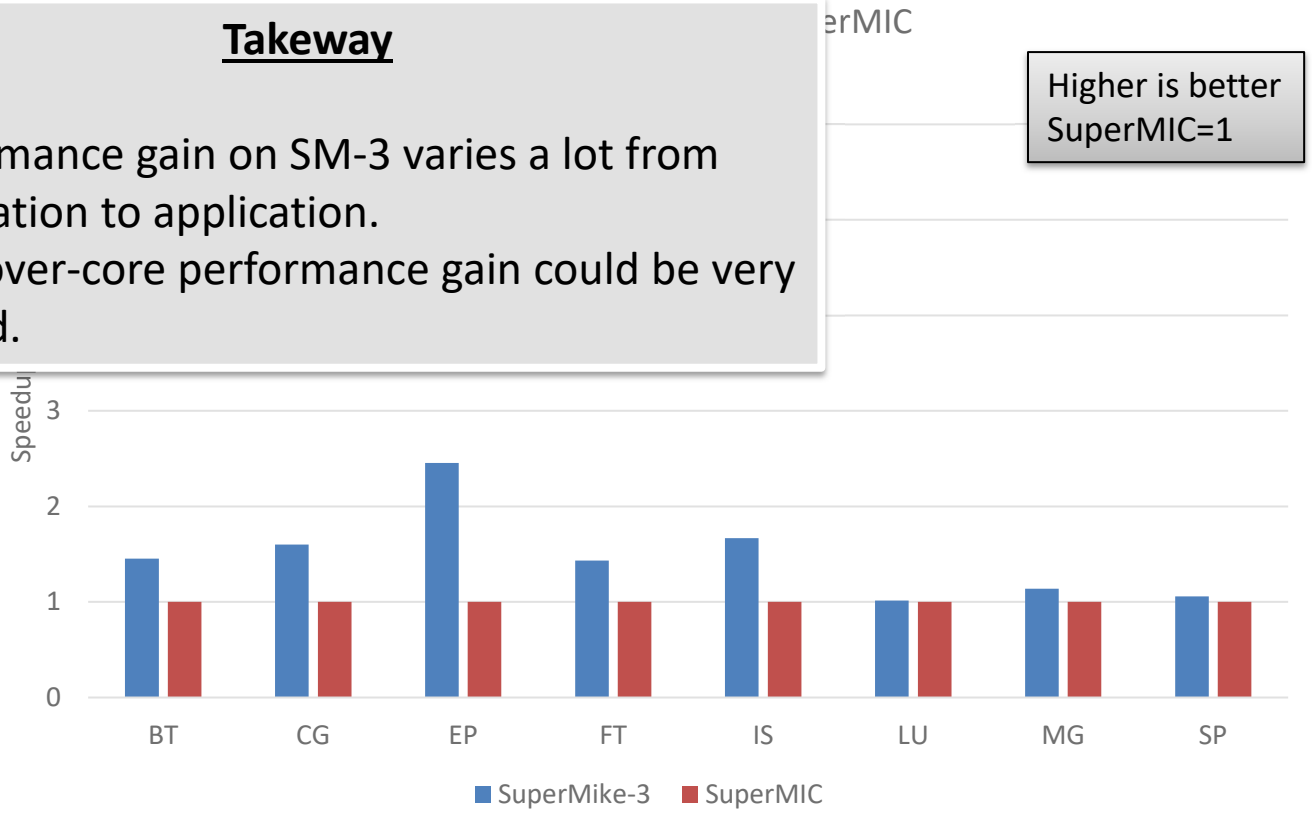


NPB 3.4.2, Class C
Intel 2021.5.1 with "-O3 –xCORE-AVX512"
KMP_AFFINITY=balanced

NPB Benchmarks – Core over Core

Takeway

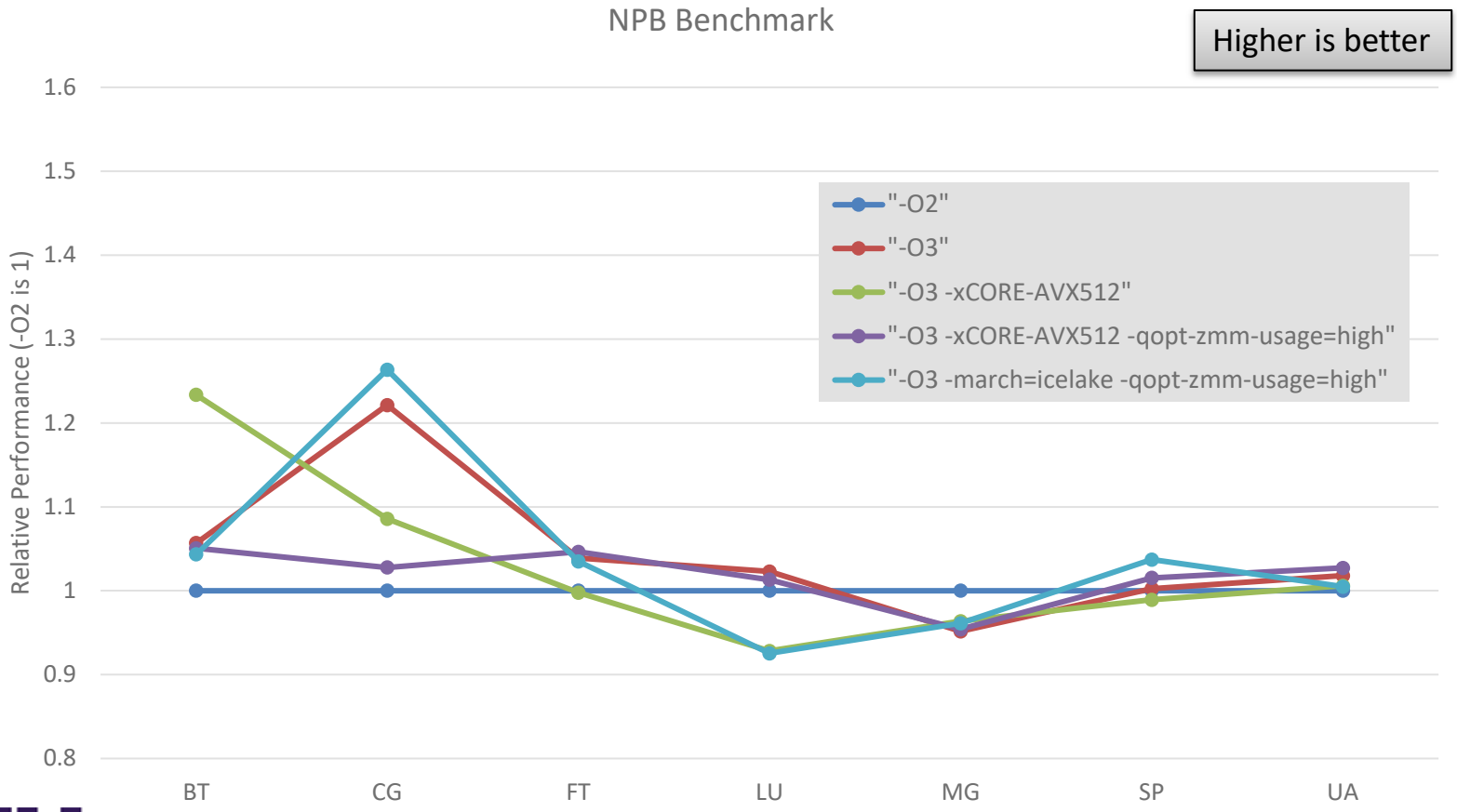
- Performance gain on SM-3 varies a lot from application to application.
- Core-over-core performance gain could be very limited.



NPB 3.4.2, Class C
Intel 2021.5.1 with "-O3 –xCORE-AVX512"
KMP_AFFINITY=balanced

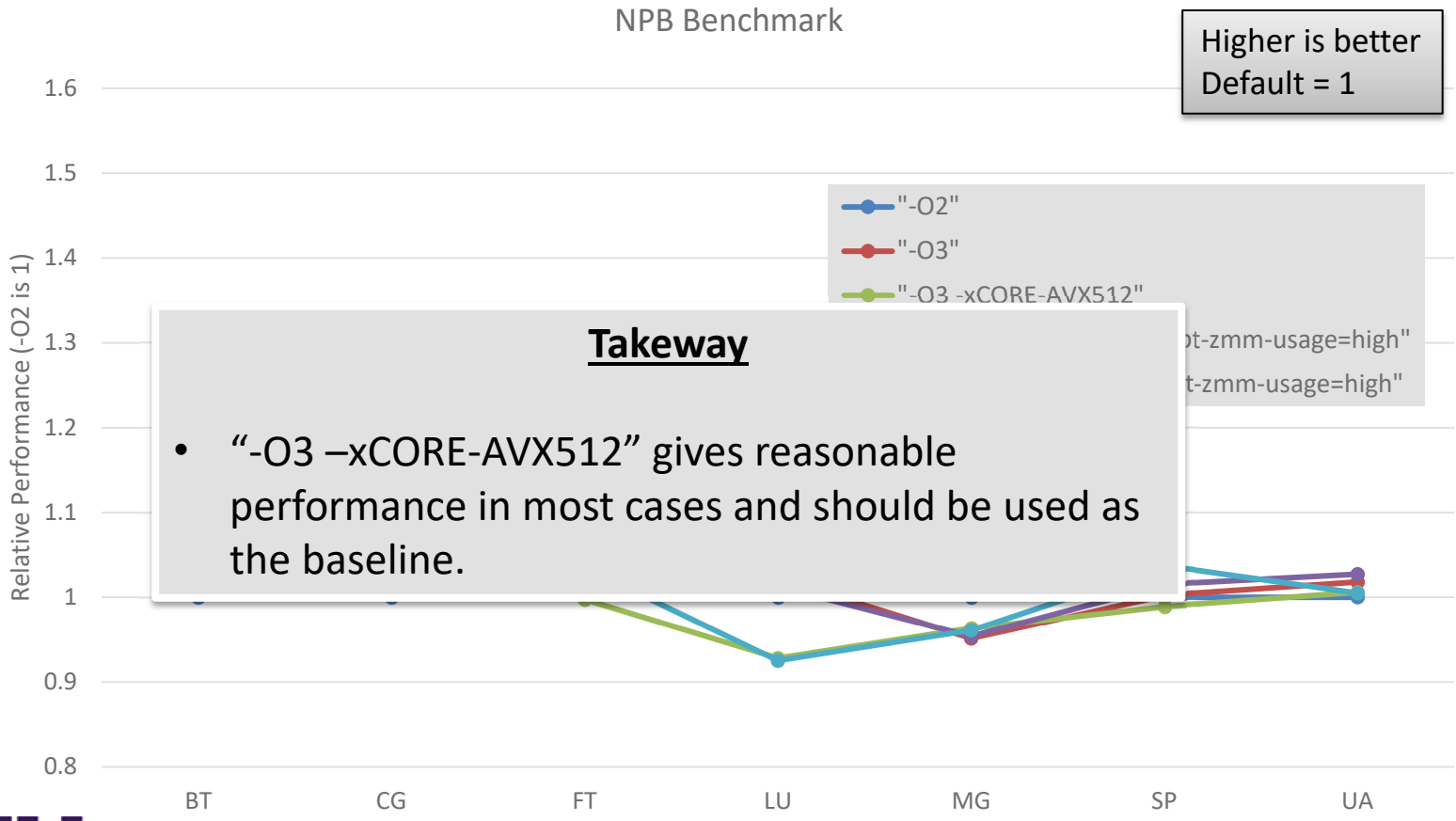
LSU CENTER FOR COMPUTATION & TECHNOLOGY

NPB Results – Compiler Flags



NPB 3.4.1, Class D
Intel 2021.5.1
KMP_AFFINITY=balance

NPB Results – Compiler Flags

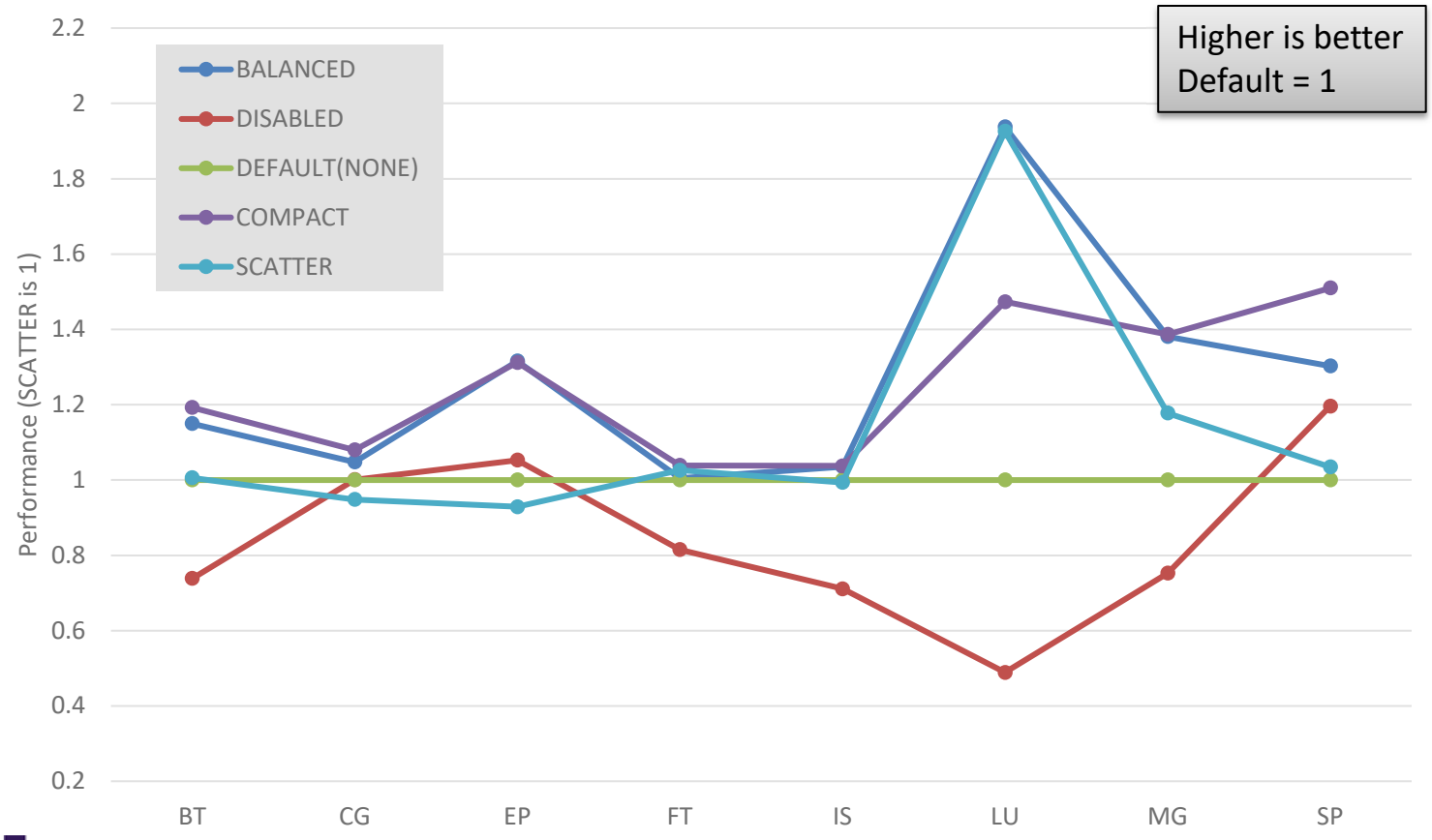


Takeway

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

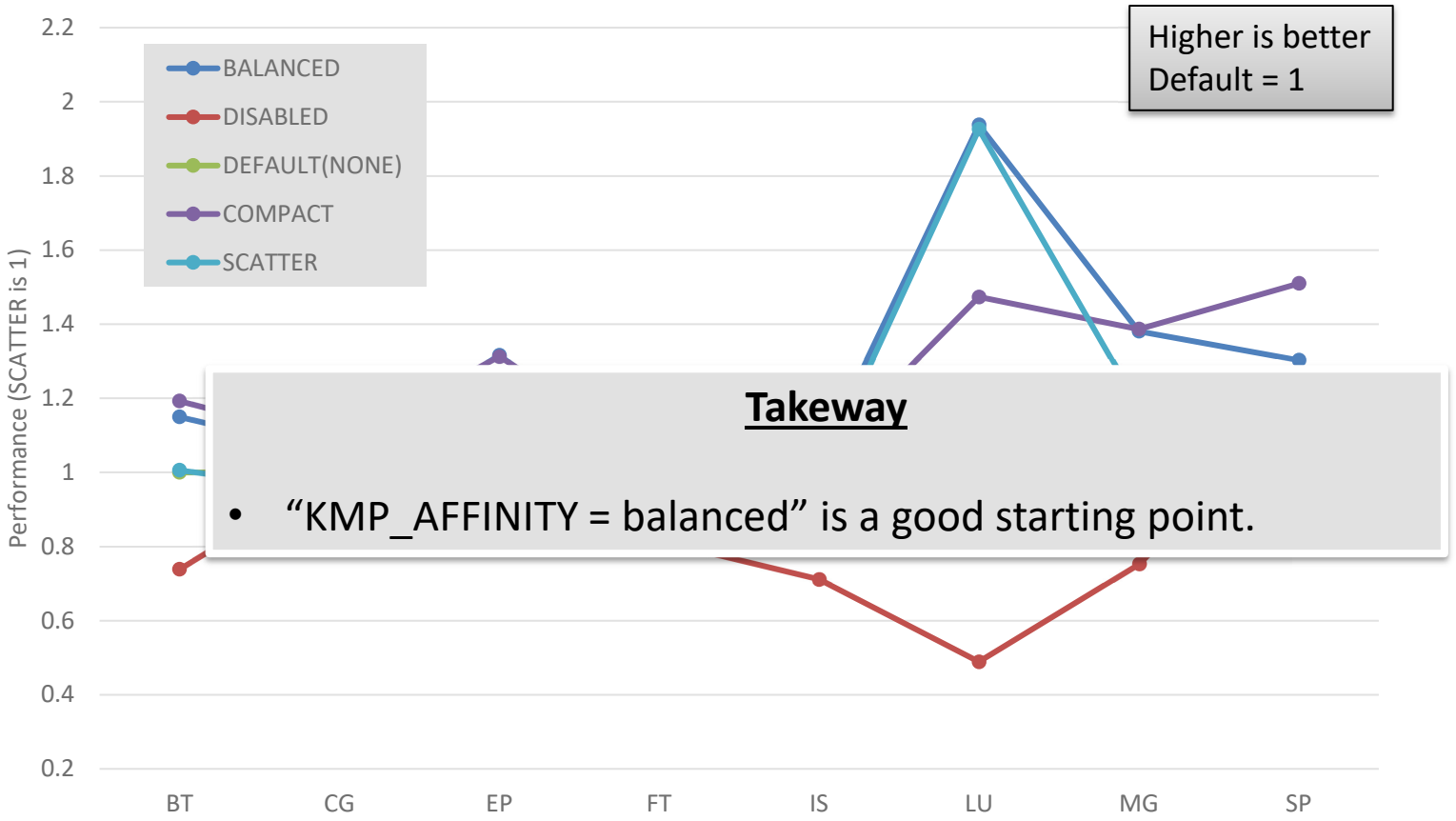
NPB 3.4.1, Class D
Intel 2021.5.1
KMP_AFFINITY=balance

NPB Results – Thread Affinity



NPB 3.4.2, Class C
Intel 2021.5.1 with "-O3 –xCORE-AVX512"

NPB Results – Thread Affinity

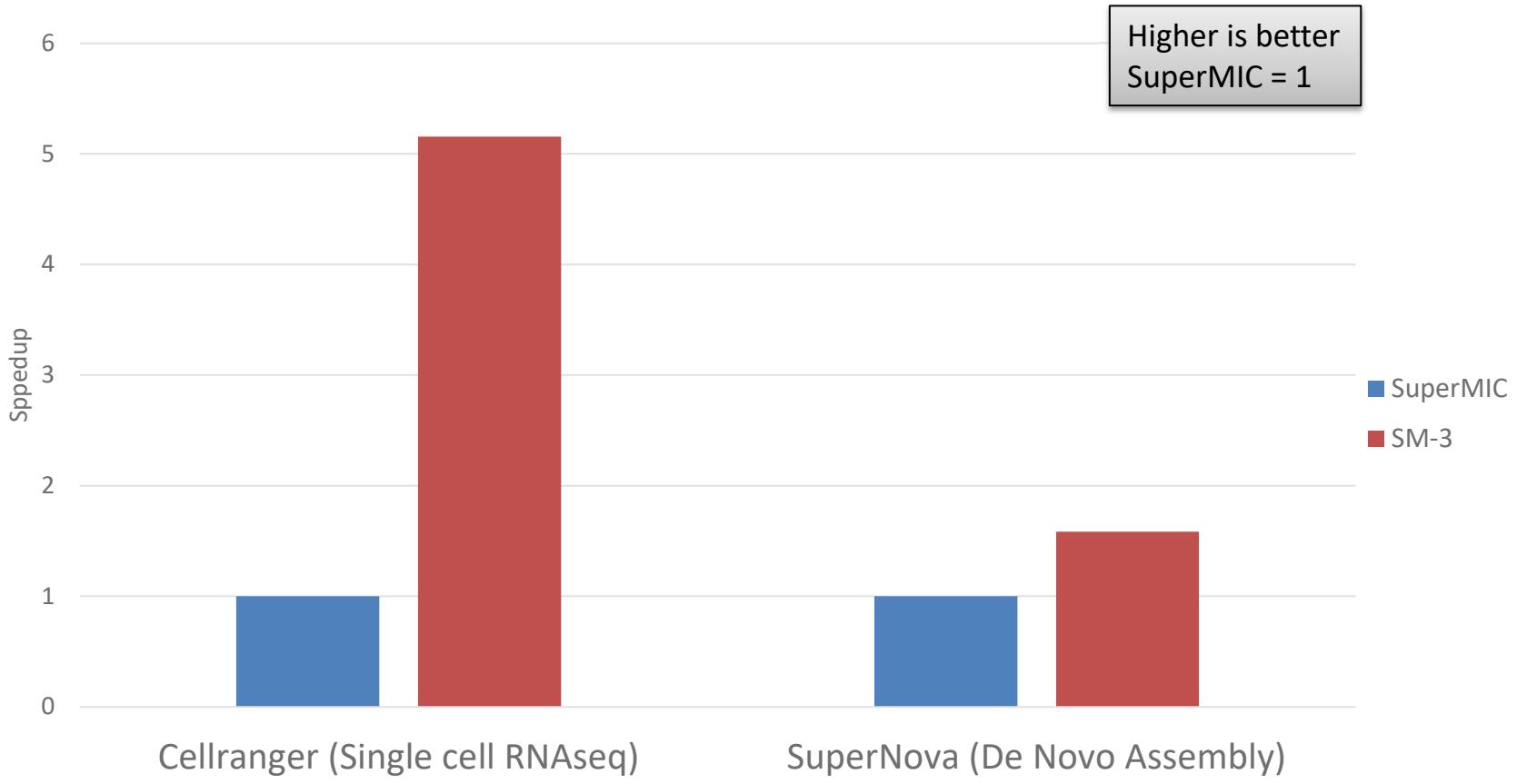


Takeway

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

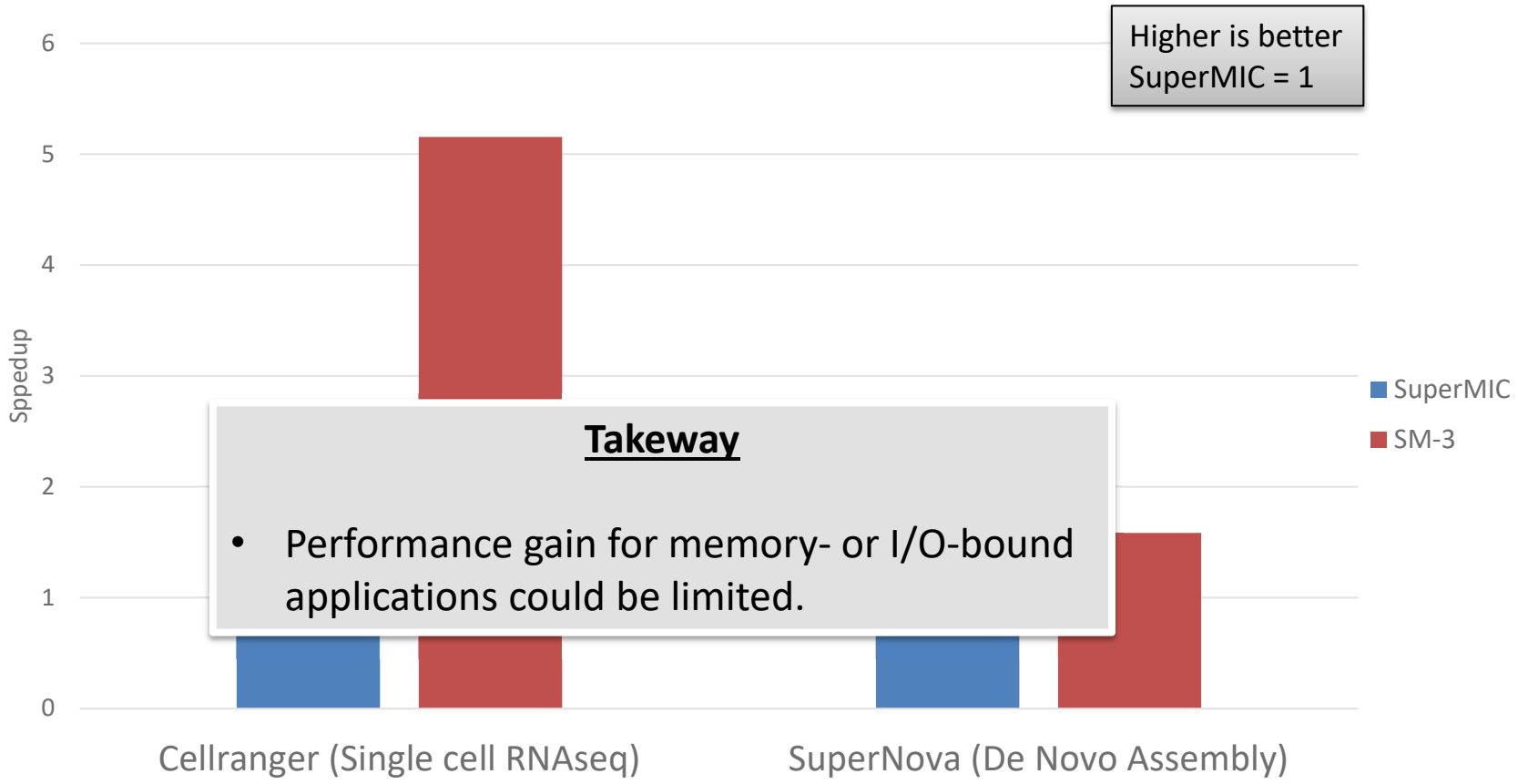
NPB 3.4.2, Class C
Intel 2021.5.1 with "-O3 –xCORE-AVX512"

Bioinformatics



Both are binary distribution from 10x Genomics
Cellranger version 7.0, mouse intestine cell, input data size = 157 GB, reference data size = 22 GB
Supernova version 2.1.1, fruit fly genome, input data size = 259 GB

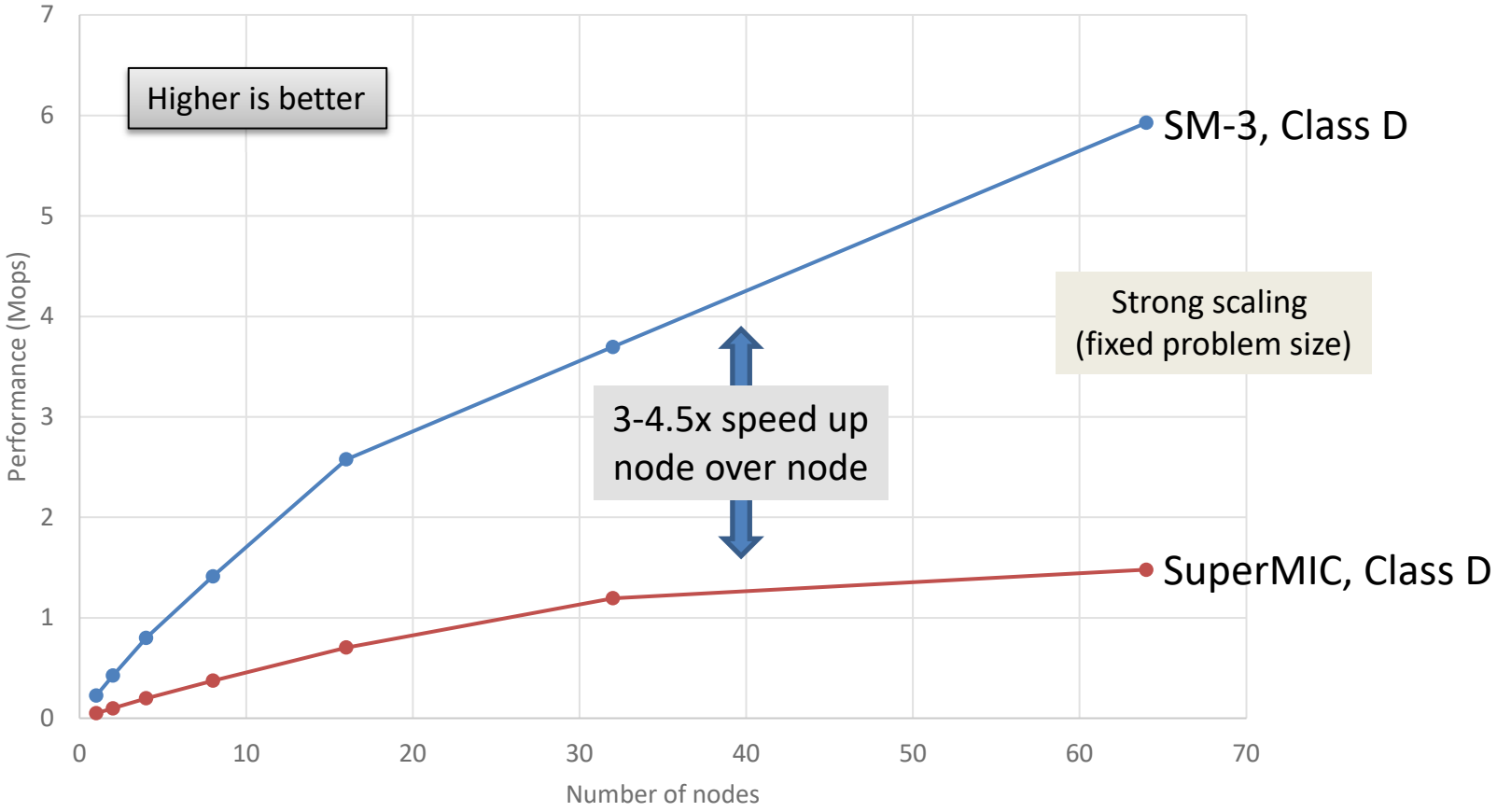
Bioinformatics



Both are binary distribution from 10x Genomics
Cellranger version 7.0, mouse intestine cell, input data size = 157 GB, reference data size = 22 GB
Supernova version 2.1.1, fruit fly genome, input data size = 259 GB

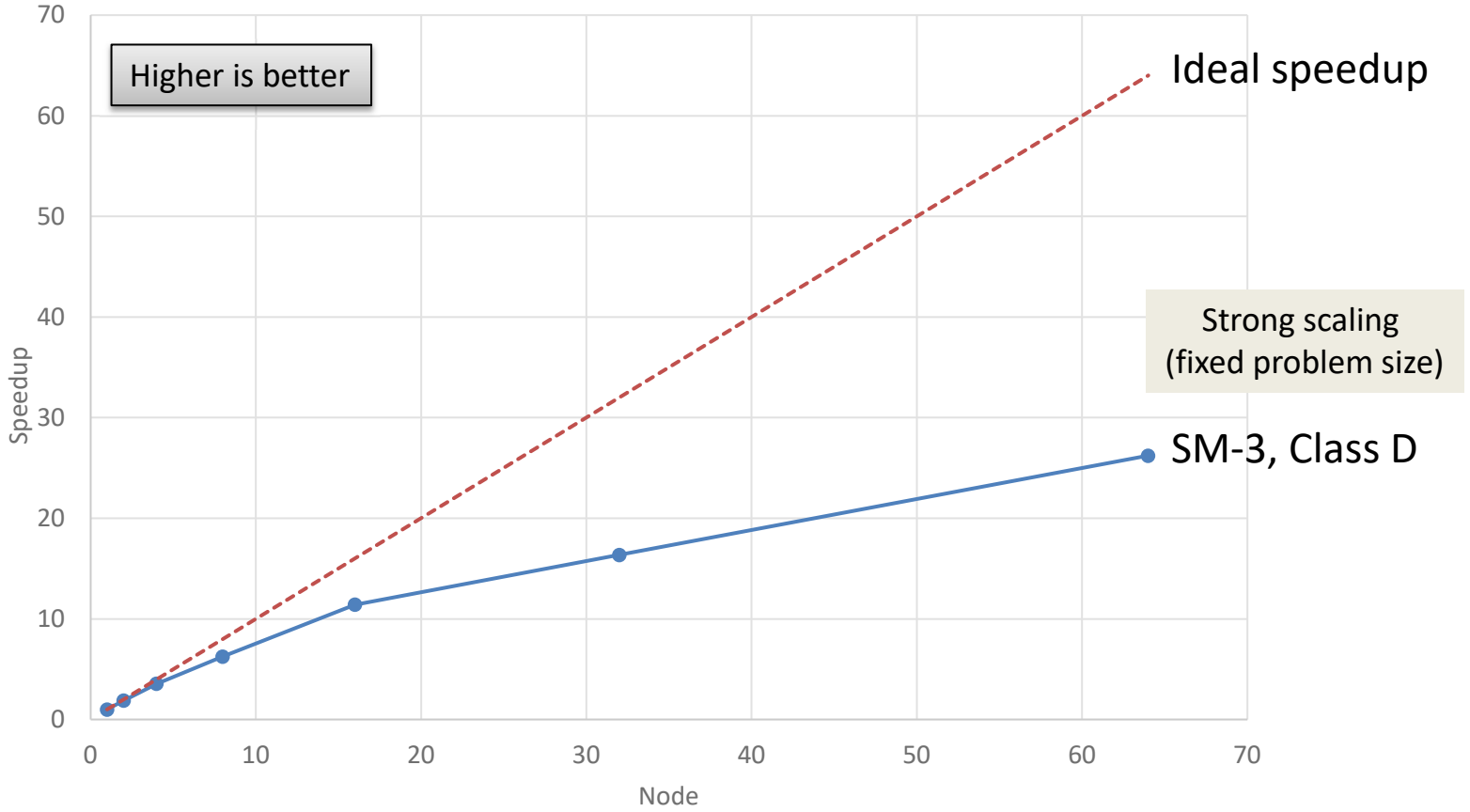
Multi-node Performance

Pure MPI - NPB LU Benchmark



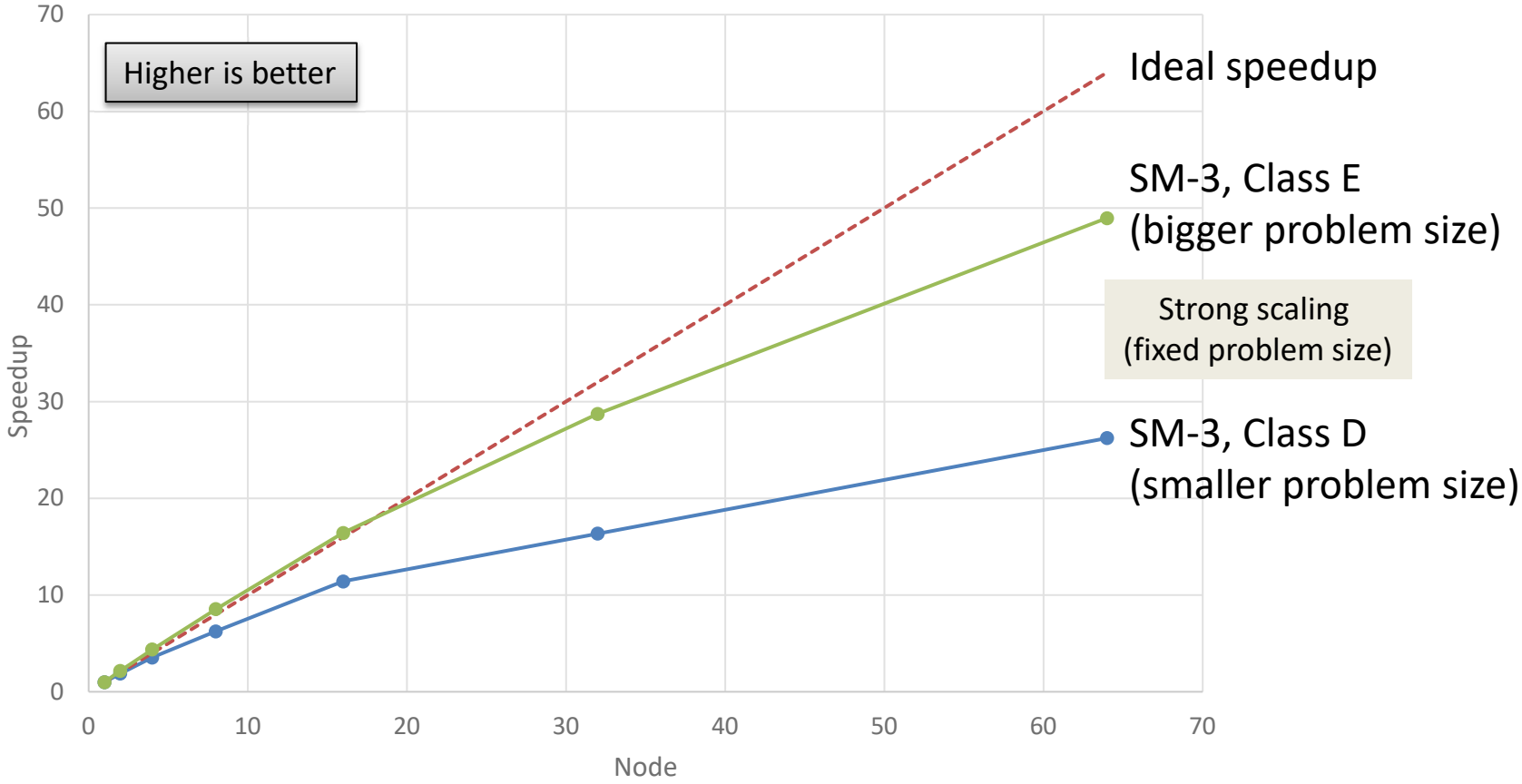
NPB 3.4.2, Class D
Intel MPI 2021.5.0
Intel 2021.5.1 with "-O3 –xCORE-AVX512"

Pure MPI - NPB LU Benchmark



NPB 3.4.2, Class D and E
Intel MPI 2021.5.0
Intel 2021.5.1 with "-O3 –xCORE-AVX512"

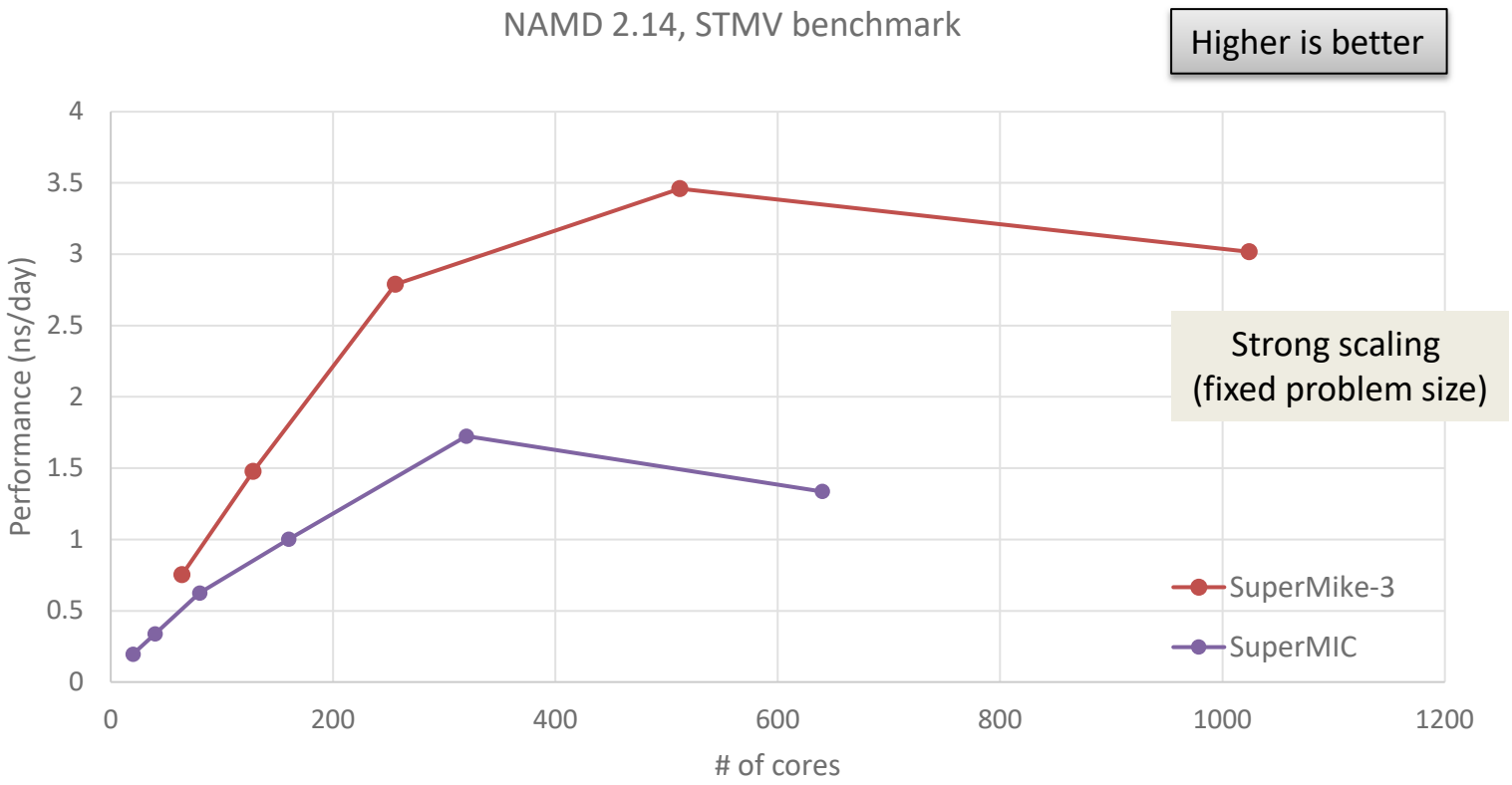
Pure MPI - NPB LU Benchmark



NPB 3.4.2, Class D and E
Intel MPI 2021.5.0
Intel 2021.5.1 with "-O3 –xCORE-AVX512"

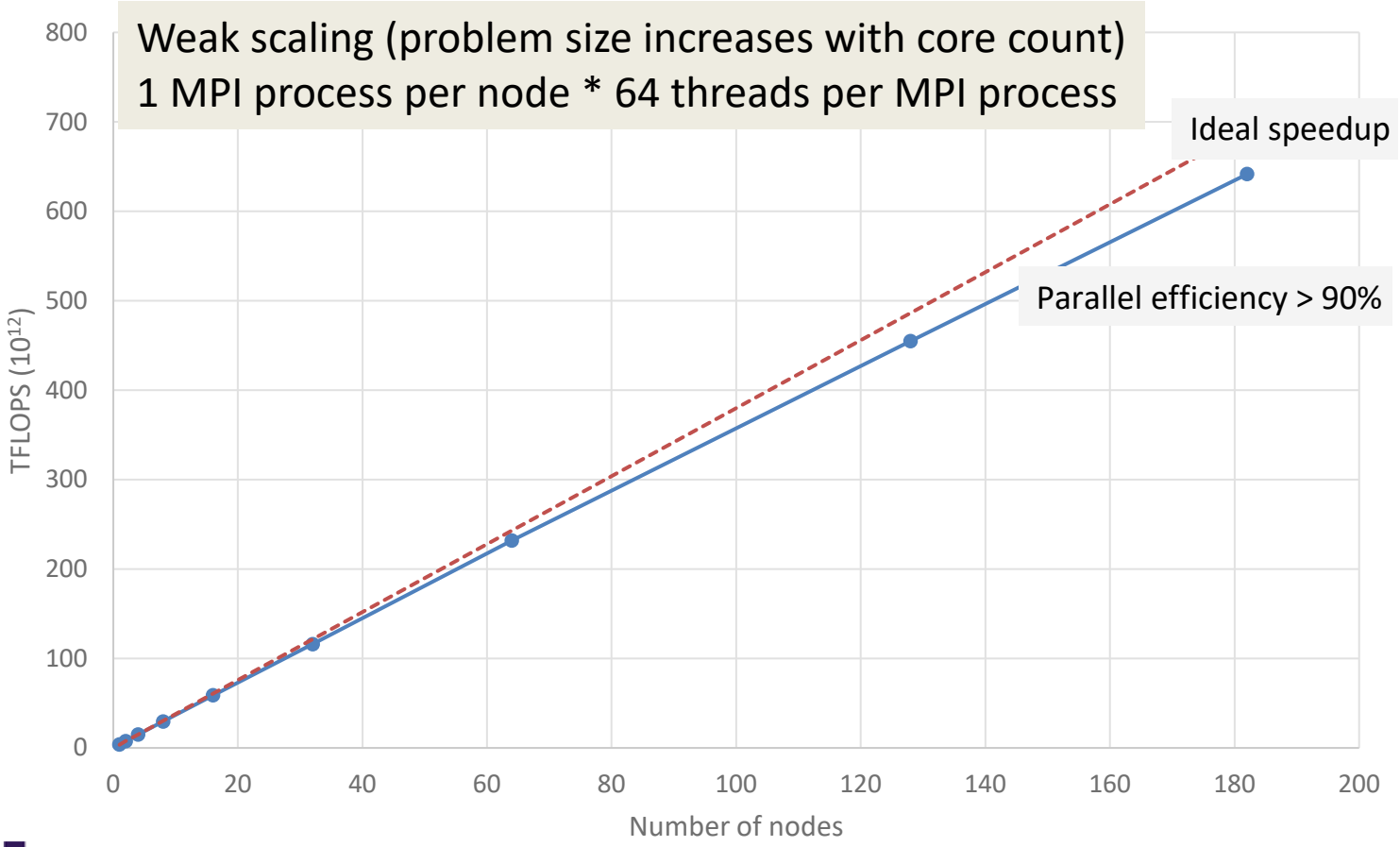
Molecular Dynamics

Pure MPI - NAMD



NAMD 2.14
Iverbs version of charmrun
Intel 2021.5.1
Intel MPI 2021.5.0

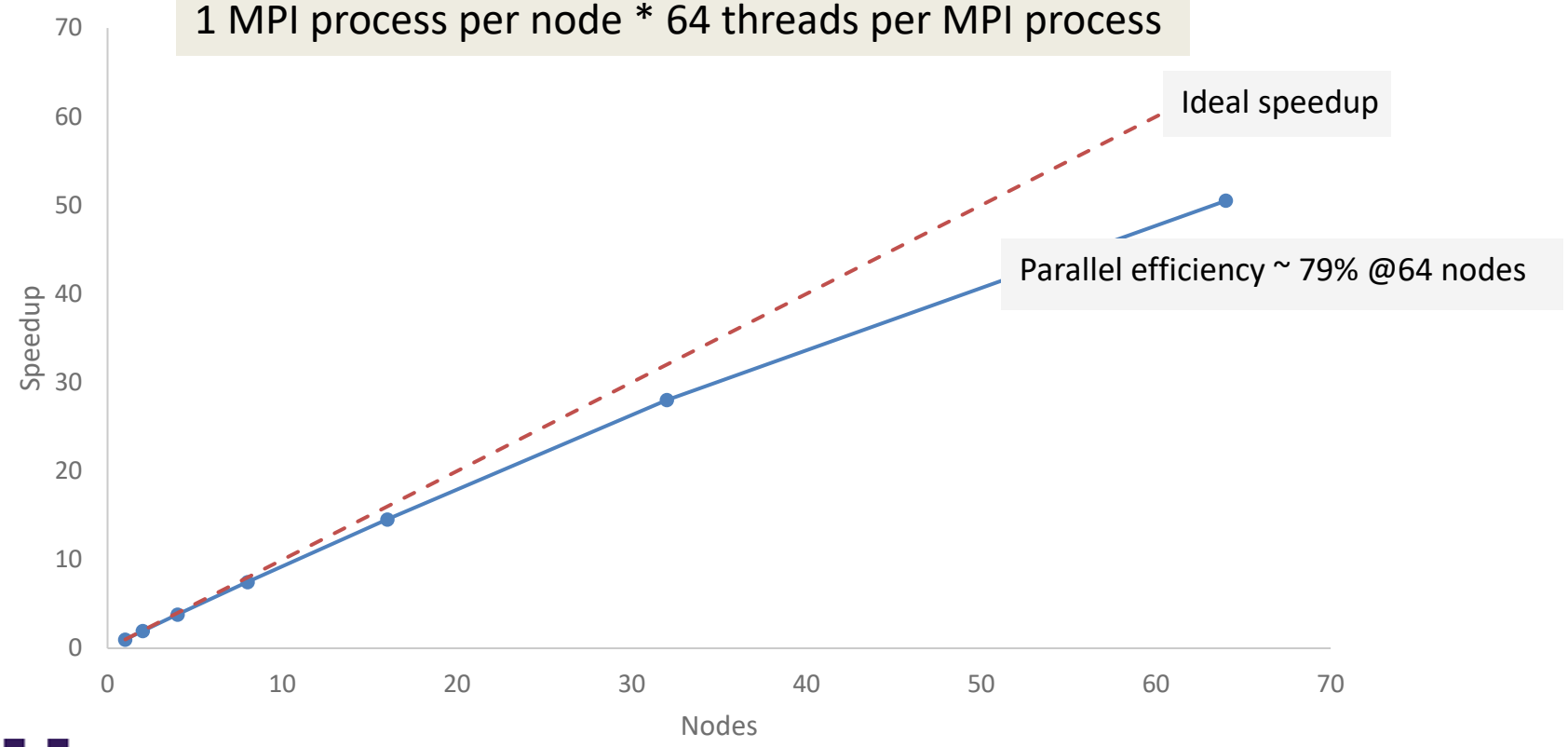
Hybrid – HPL



HPL 2.3 (Intel MKL version)
Problem size = 90% installed memory
Intel MPI 2021.5.0

Hybrid - HPCG

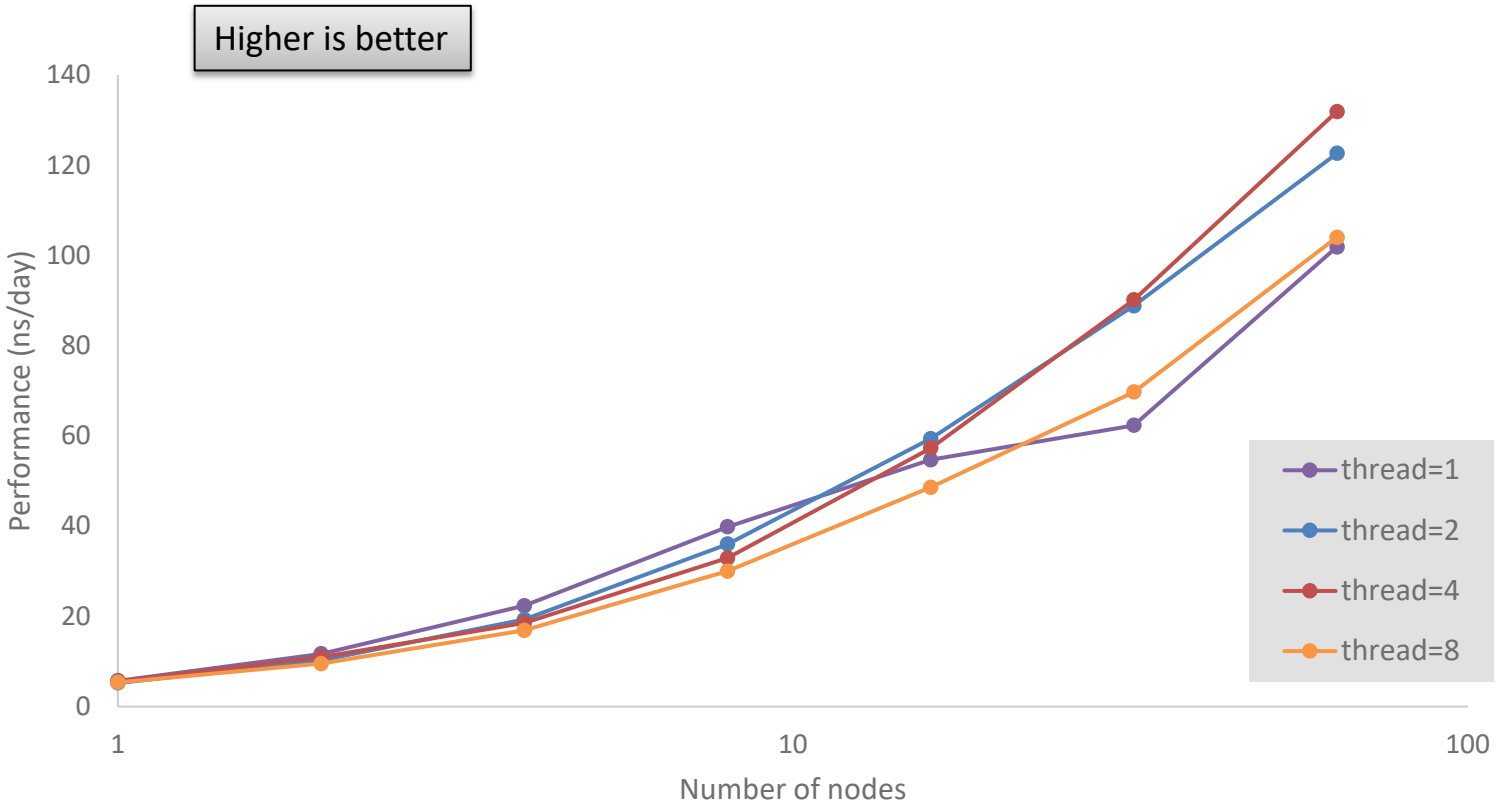
Weak scaling (problem size increases with core count)
1 MPI process per node * 64 threads per MPI process



HPCG 3.0 (Intel MKL version)
Problem size = 192 192 192
Intel MPI 2021.5.0

LSU CENTER FOR COMPUTATION & TECHNOLOGY

Hybrid - GROMACS



GROMACS 2021.3
Intel MPI 2021.5..0
Intel 2021.5.1 with "-O3 –xCORE-AVX512"

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 most cases, your application will run faster and scale better on SM-3 compared to SuperMIC
- That being said, how much faster depends on a lot of factors
- You need to figure that out before making a (educated) decision whether switch to SM-3
- You need to run your own experiments

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
- 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, which depends on applications as well as problem size

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, which depends on applications as well as problem size

If you need help, let us know!