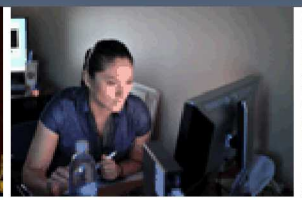
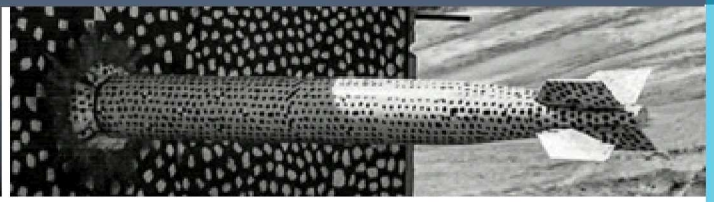
SAND2019-2378PE

Astra (ARM) Microbenchmarks





PRESENTED BY

Douglas M. Pase, PhD/Sandia National Laboratories

With Anthony M. Agelastos and Joel O. Stevenson

SAND2019-XXXX PE

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

All data are from gathered from unclassified systems using commercially available, non-proprietary hardware.

All benchmarks are simple variants of common benchmarks and/or public libraries (DAXPY, DGEMM, STREAM, MPI_Sendrecv, etc.)

Contents

- Processor Description
- DAXPY - Cache and memory throughput (vector and scalar)
- DGEMM - Vector floating-point performance
- Stream - Memory throughput
- Gather - Indirect loads
- Scatter - Indirect stores
- MPI_Sendrecv and MPI_Allreduce - Kernel transition and communication performance

Cavium ARM (ThunderX2)

- 16 nm lithography
- 28 cores per socket
- 2.0 GHz - 2.5 GHz clock (?)
- 28 MB L3/socket (1 MB/core)
- *Unknown* GT/s inter-chip ICN
- 2666 MHz DDR4
- 8 memory channels/socket
- Advanced SIMD extension

- 32 KiB L1 instruction cache/core
- 32 KiB L1 data cache/core
- 256 KiB L2 combined cache/core
- 28 MB L3/socket (1 MB/core)
- *Unknown* ITLB/DTLB sizes
- 2048-entry STLB (4 KiB pages)

DAXPY

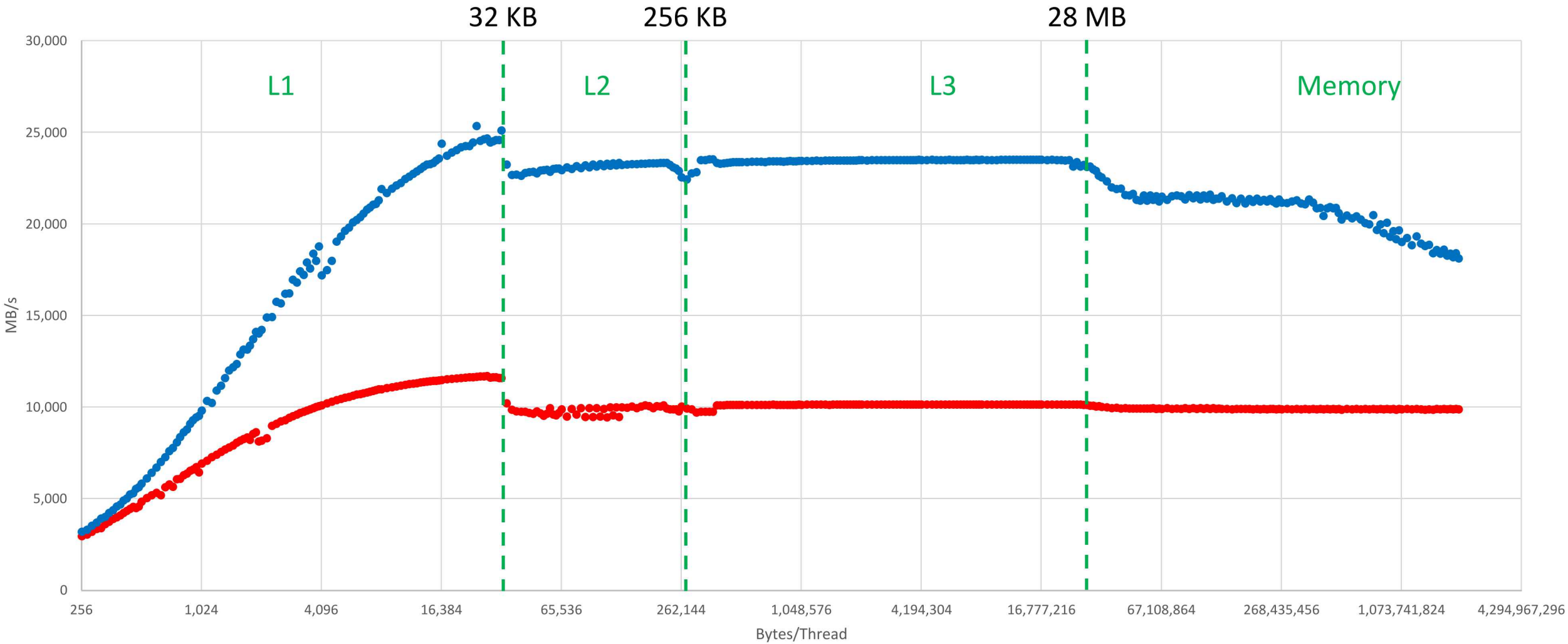
`y[i] = α*x[i] + y[i]`

x, **y** are double-precision floating-point vectors

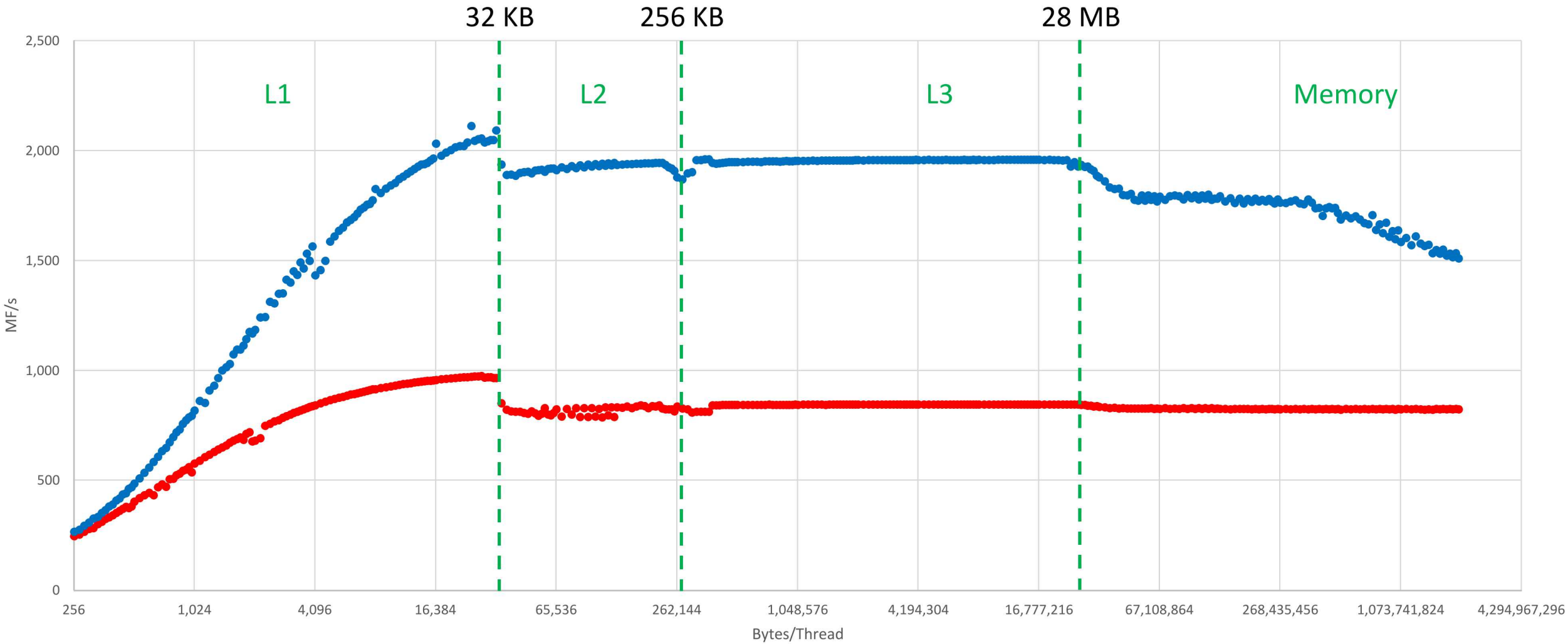
α is a double-precision floating-point scalar

i is an integer index; * is the scalar multiplication operator

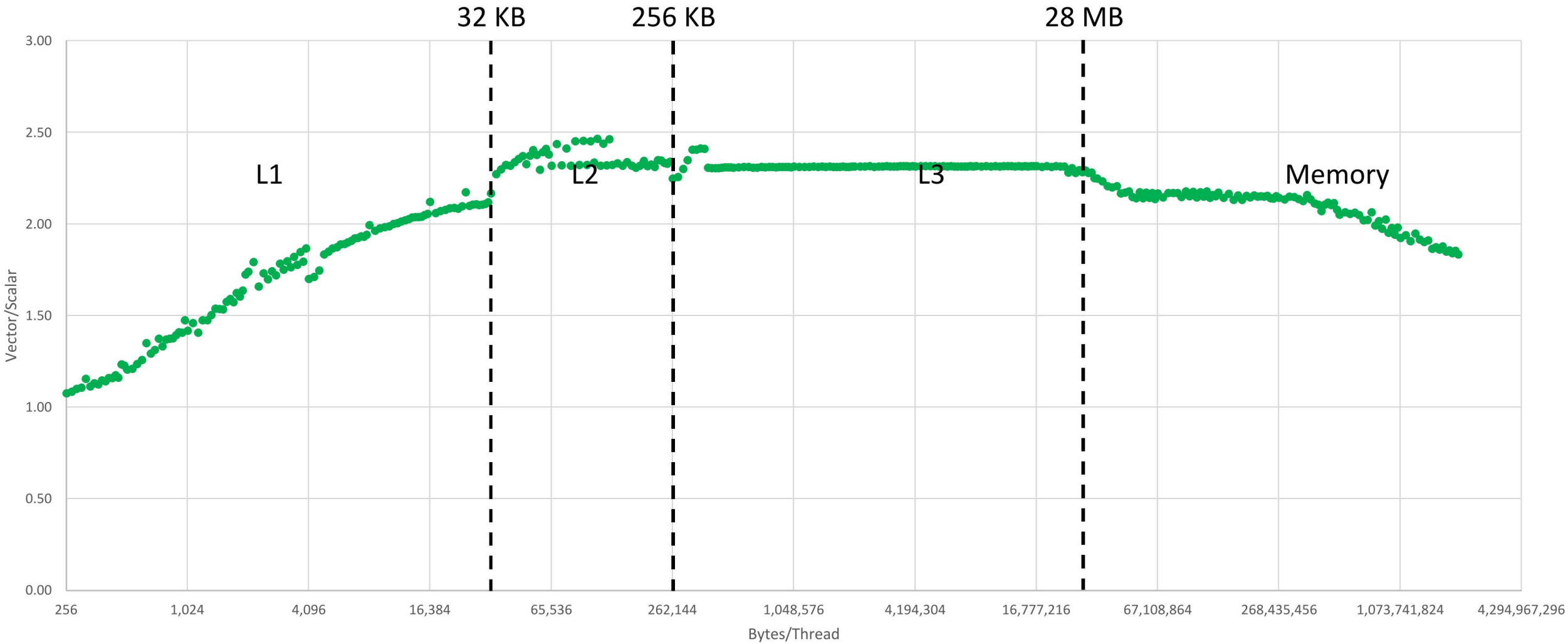
DAXPY Memory Throughput (One Core)



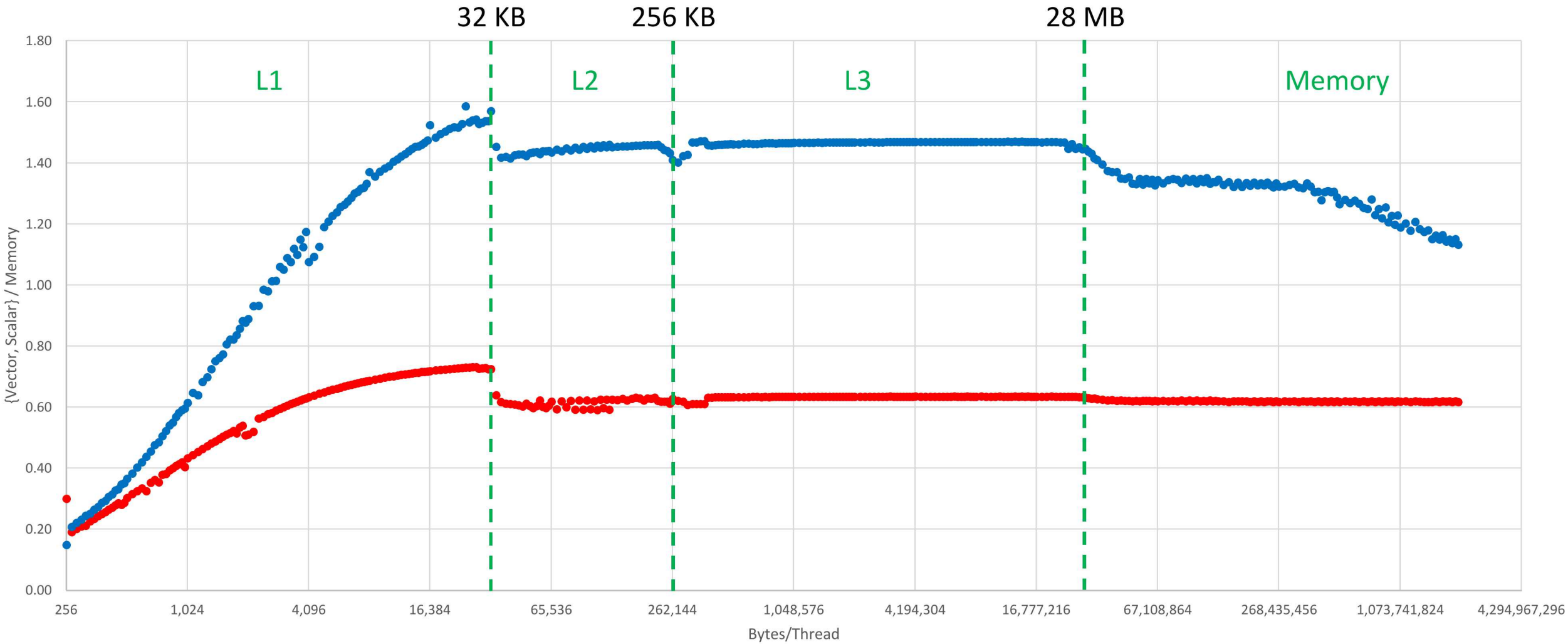
DAXPY Floating-Point Throughput (One Core)



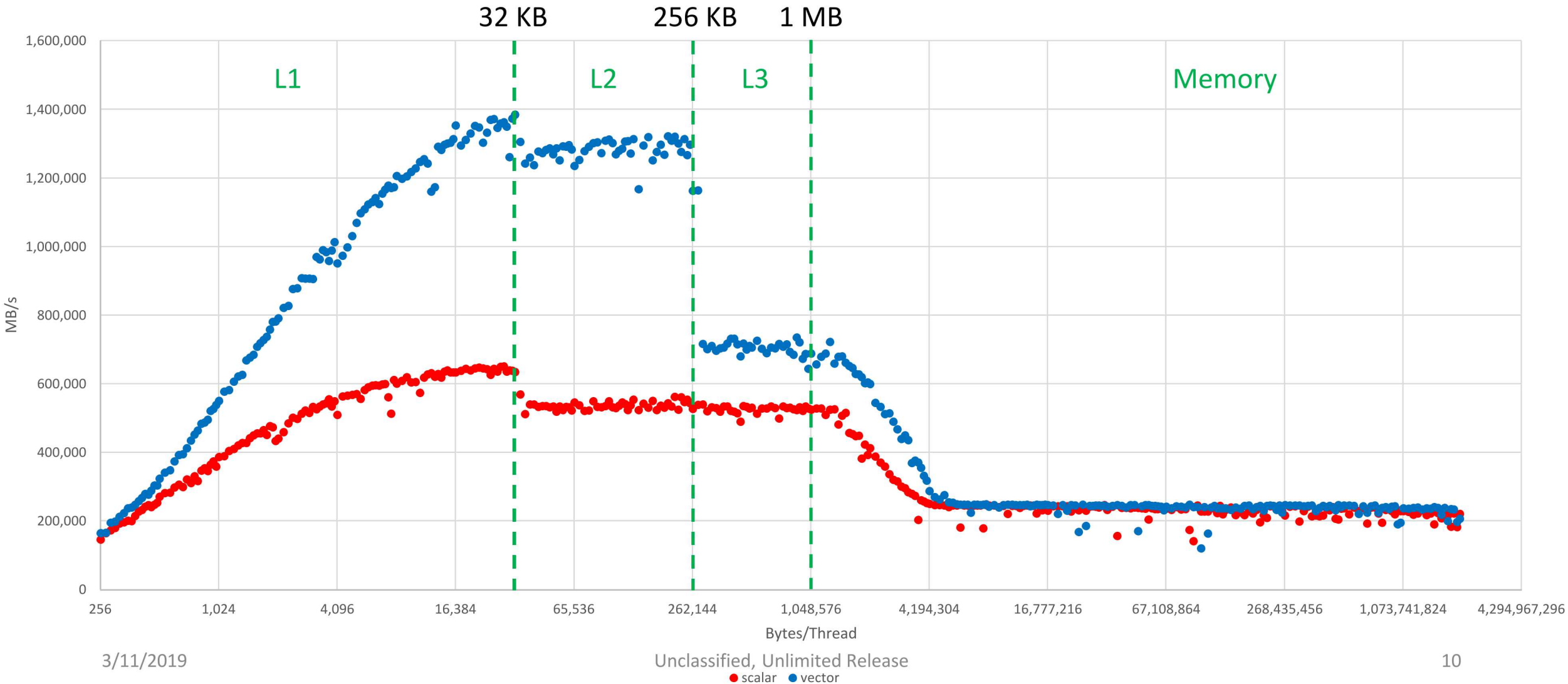
DAXPY Vector/Scalar (One Core)



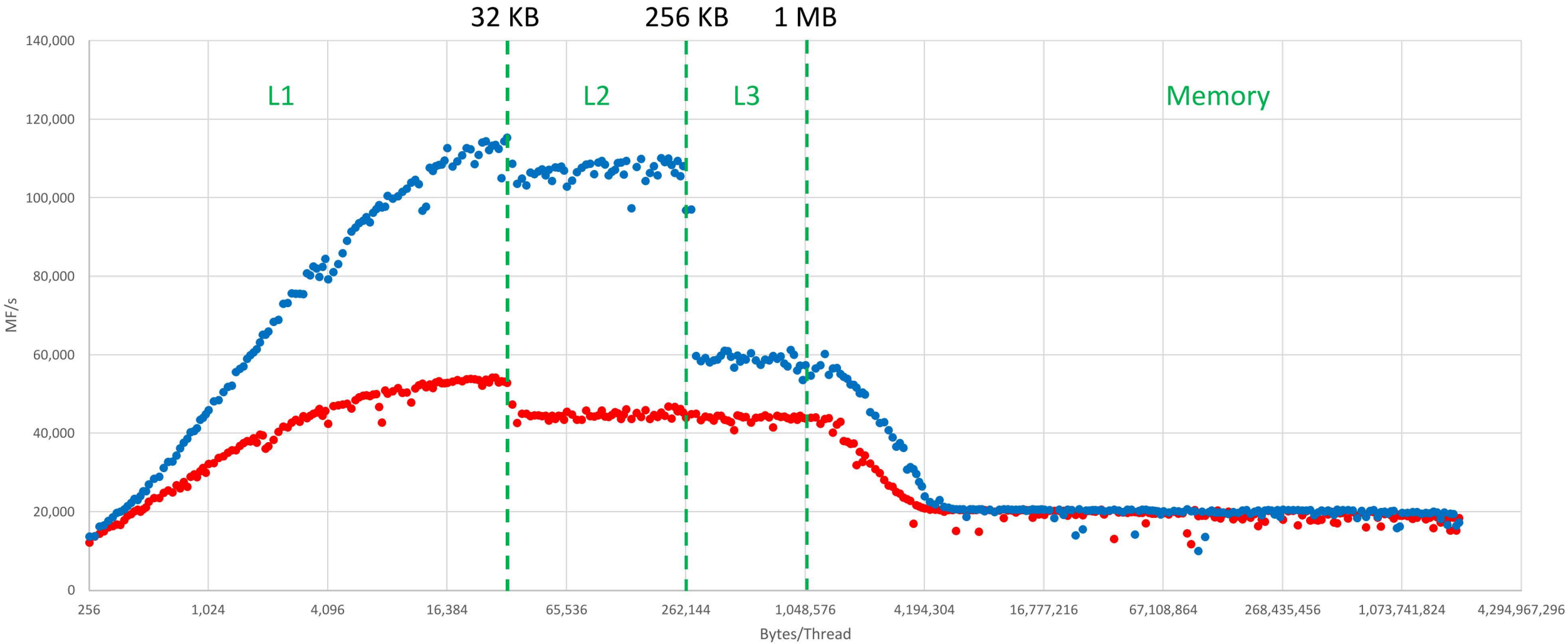
Performance Relative to Memory (One Core)



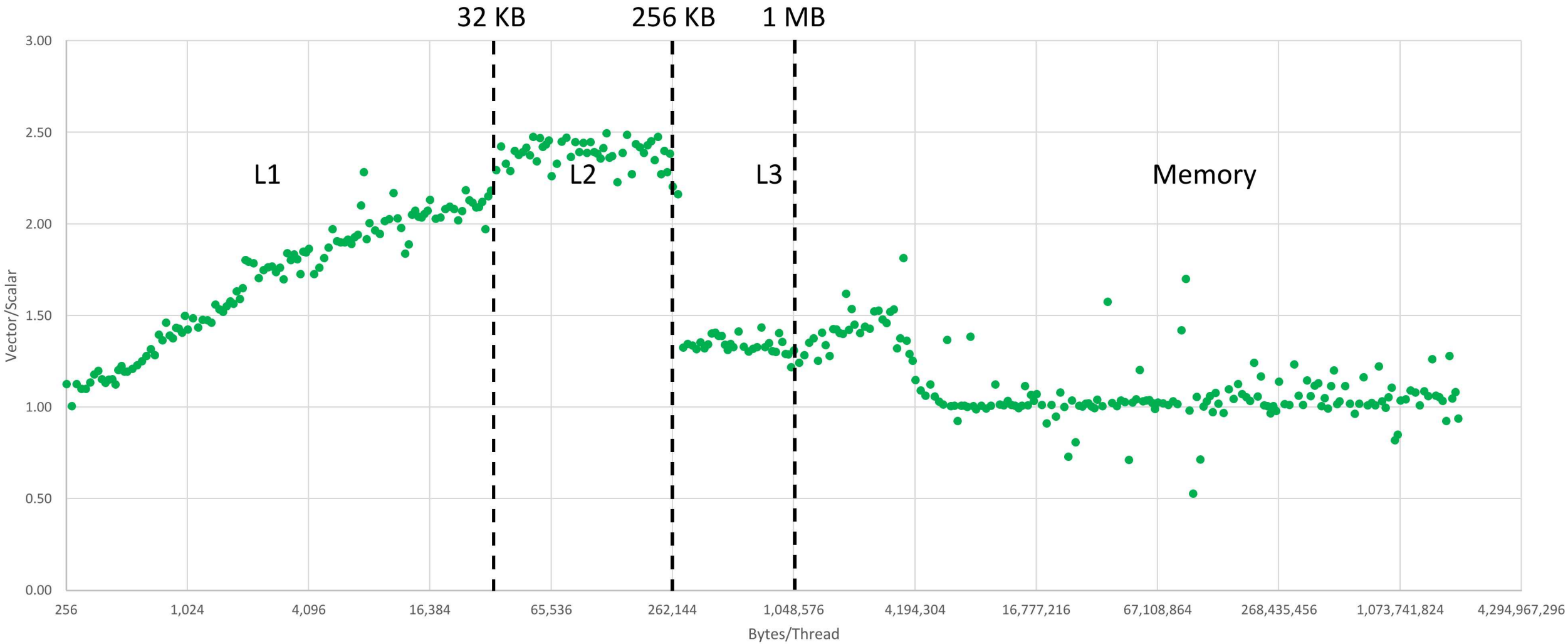
DAXPY Memory Throughput (56 Cores)



DAXPY Floating-Point Throughput (56 Cores)



DAXPY Vector/Scalar (56 Cores)



Performance Relative to Memory (56 Cores)

DGEMM

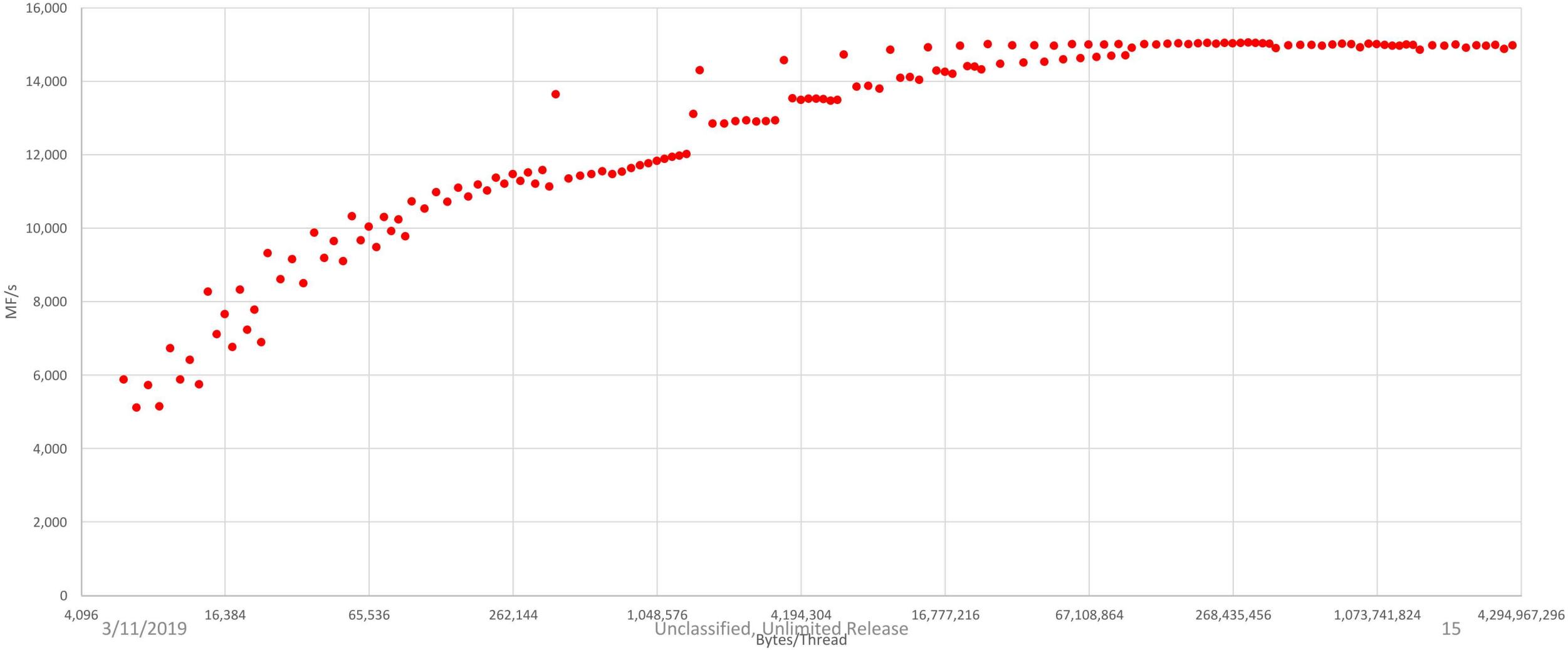
$\underline{C} = \alpha * (\underline{A} \times \underline{B}) + \beta * \underline{C}$

$\underline{A}$, $\underline{B}$, $\underline{C}$ are double-precision floating-point matrices

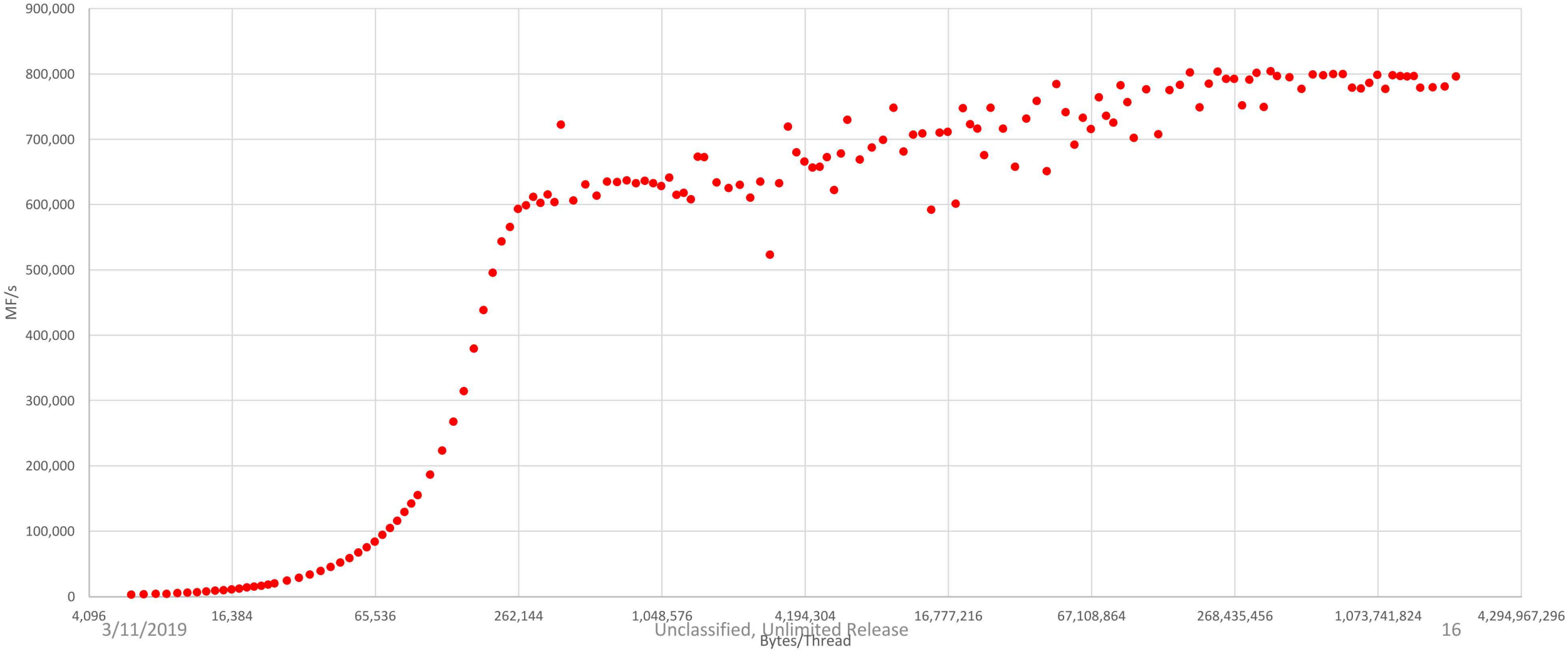
α, β are double-precision floating-point scalars

$*$ is the scalar multiplication operator; x is the matrix multiplication operator

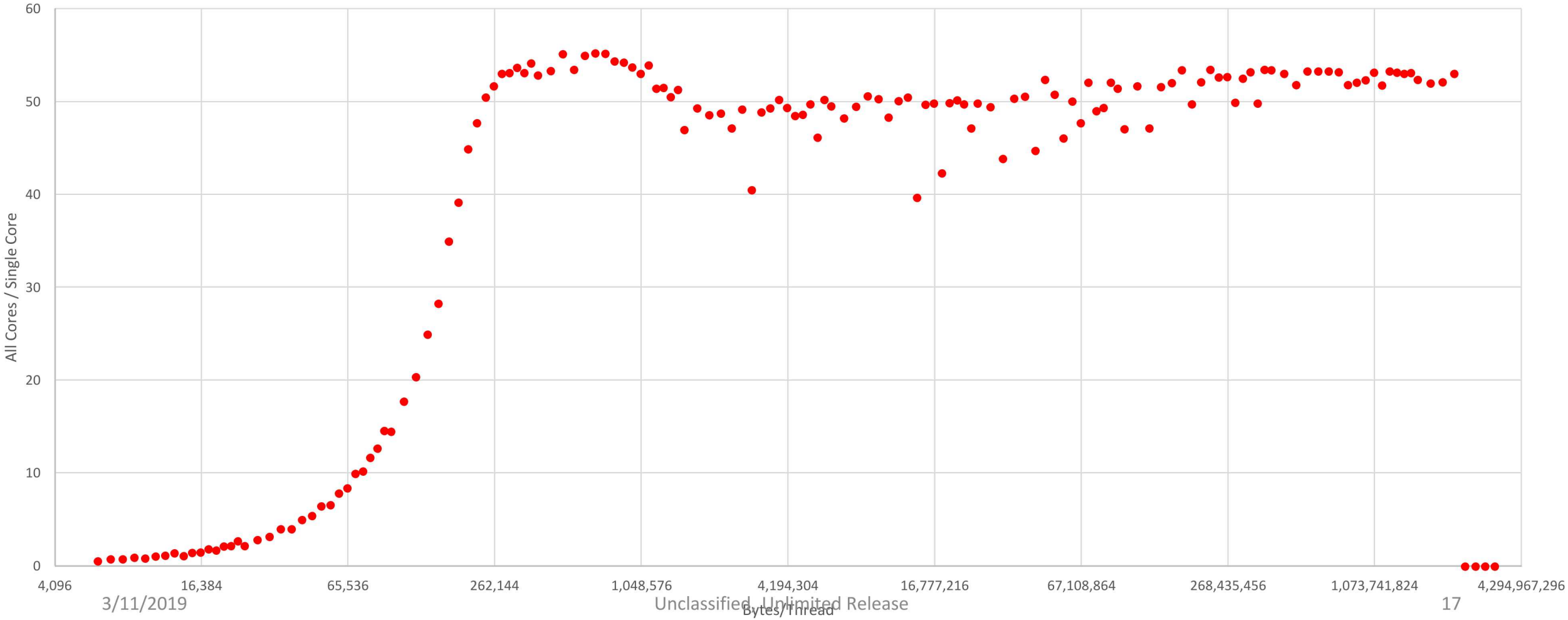
DGEMM MF/s (One Core)



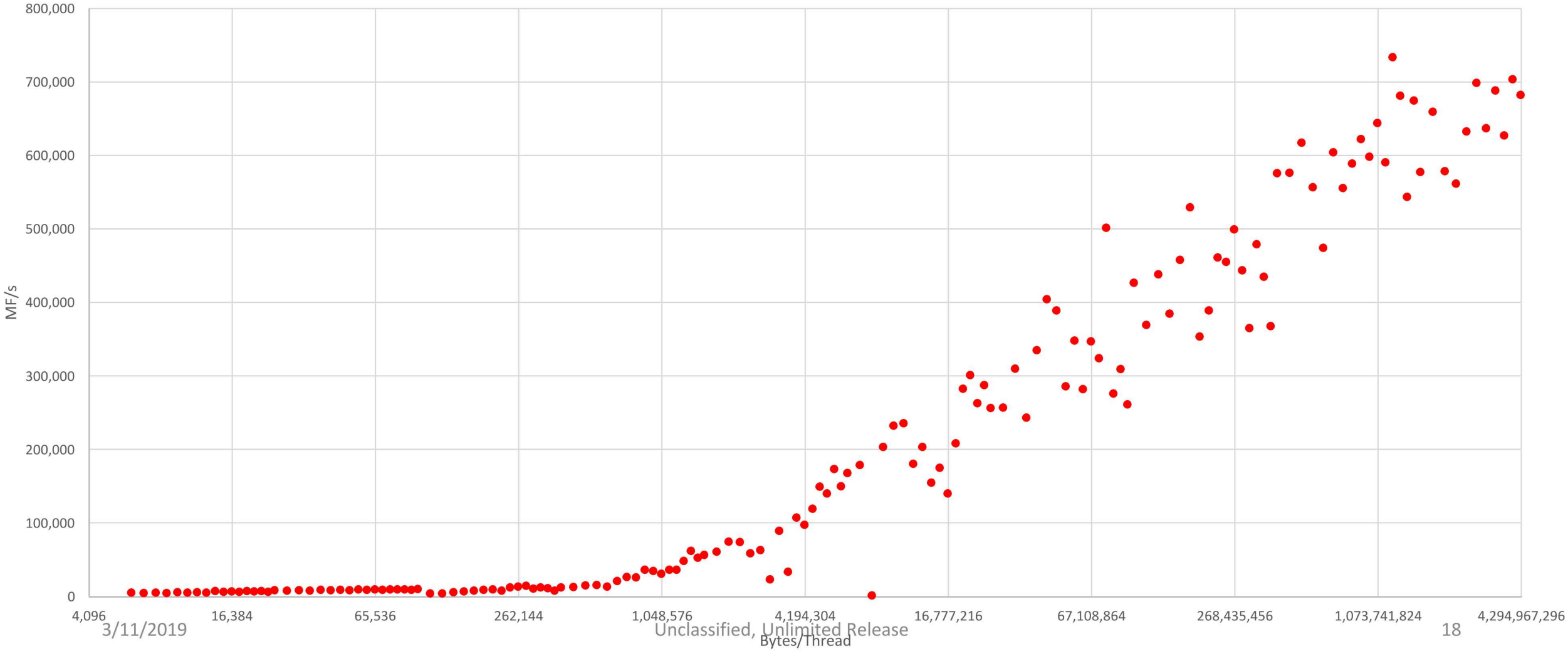
DGEMM MF/s (56 Cores)



Ratio of 56 Cores / 1 Core



OMP Version (56 Cores)



Stream

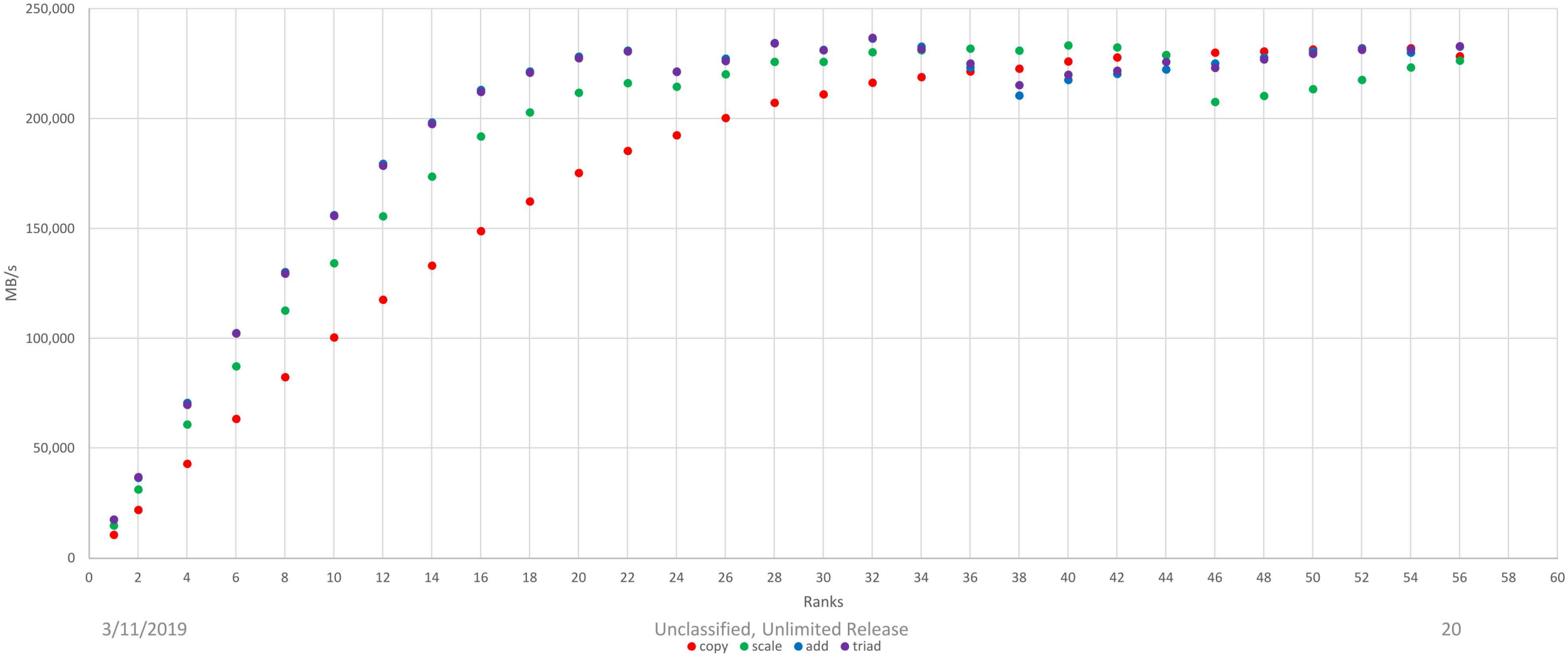
Copy: `y[i] = x[i]`

Scale: `y[i] = α*x[i]`

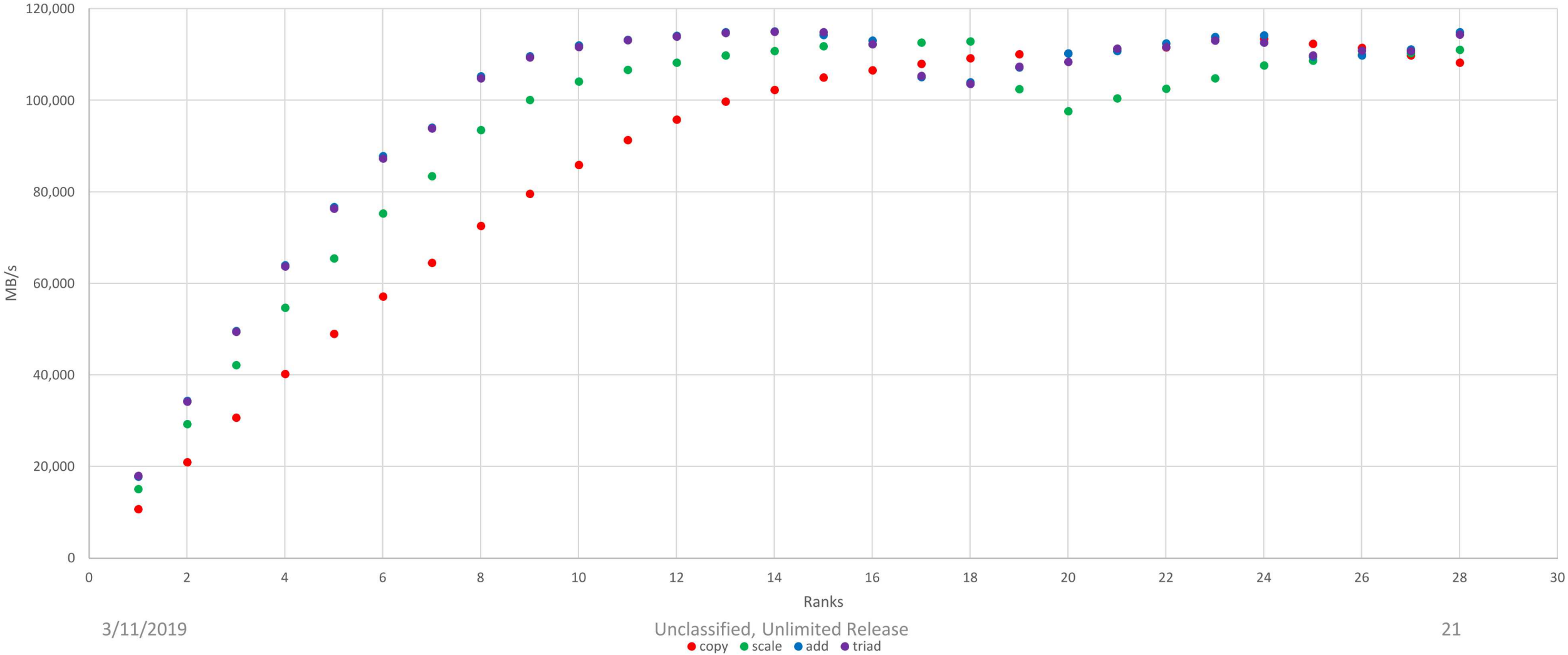
Add: `z[i] = x[i] + y[i]`

Triad: `z[i] = x[i] + α*y[i]`

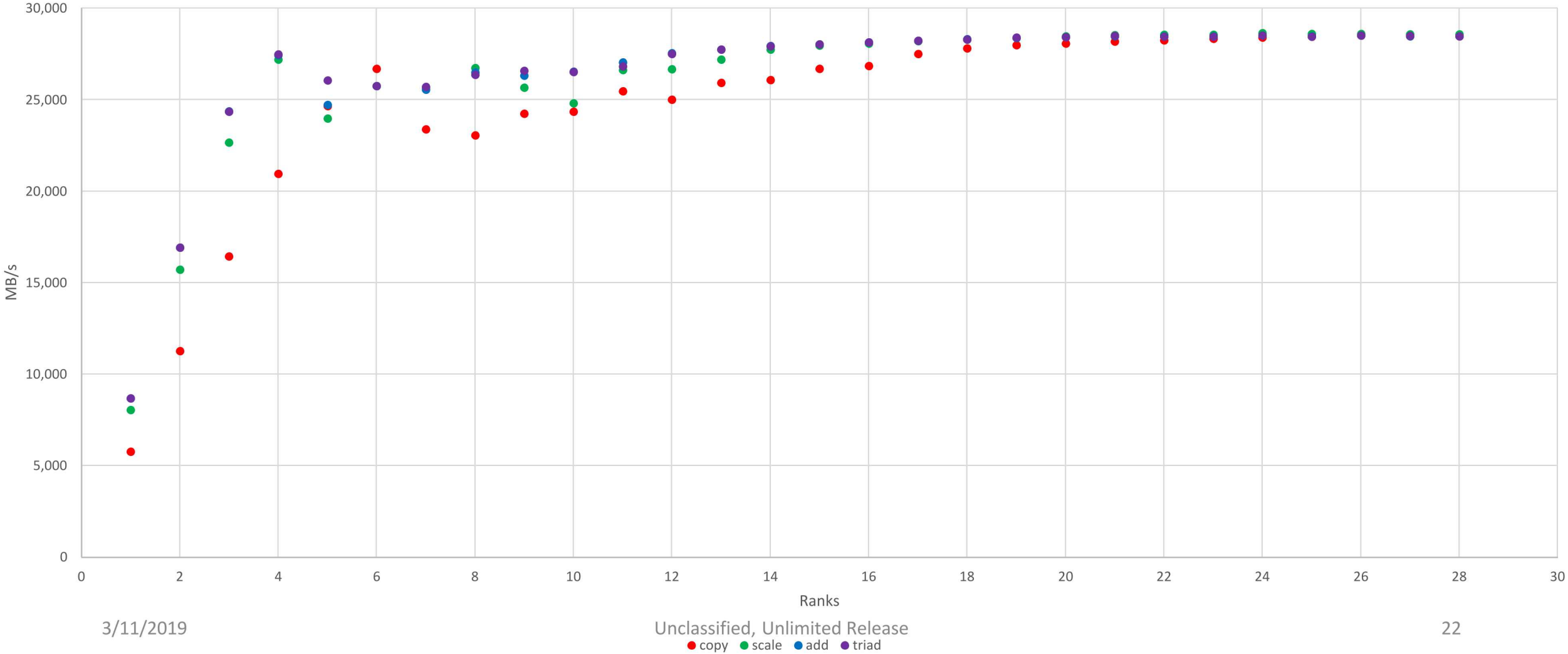
Stream Memory Throughput (2 Sockets)



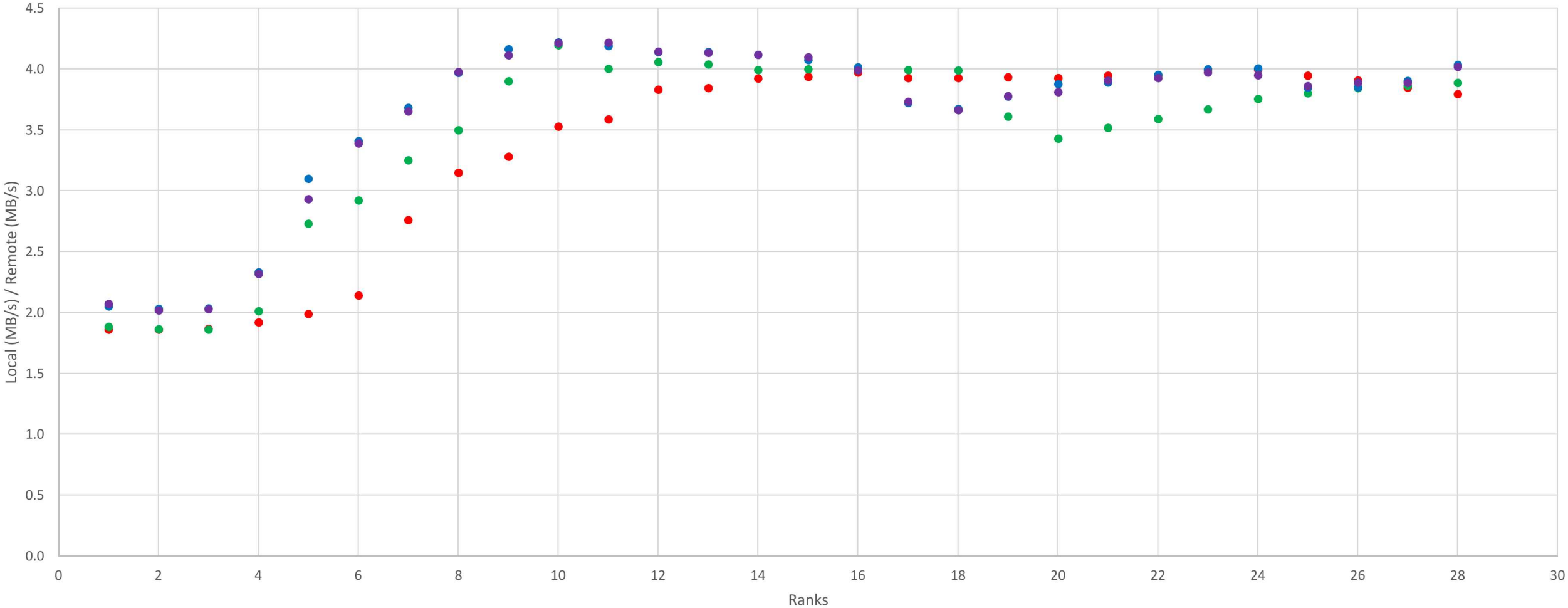
Local Memory Throughput (1 Socket)



Remote Memory Throughput (1 Socket)



Local vs. Remote Memory B/W (1 Socket)



Gather

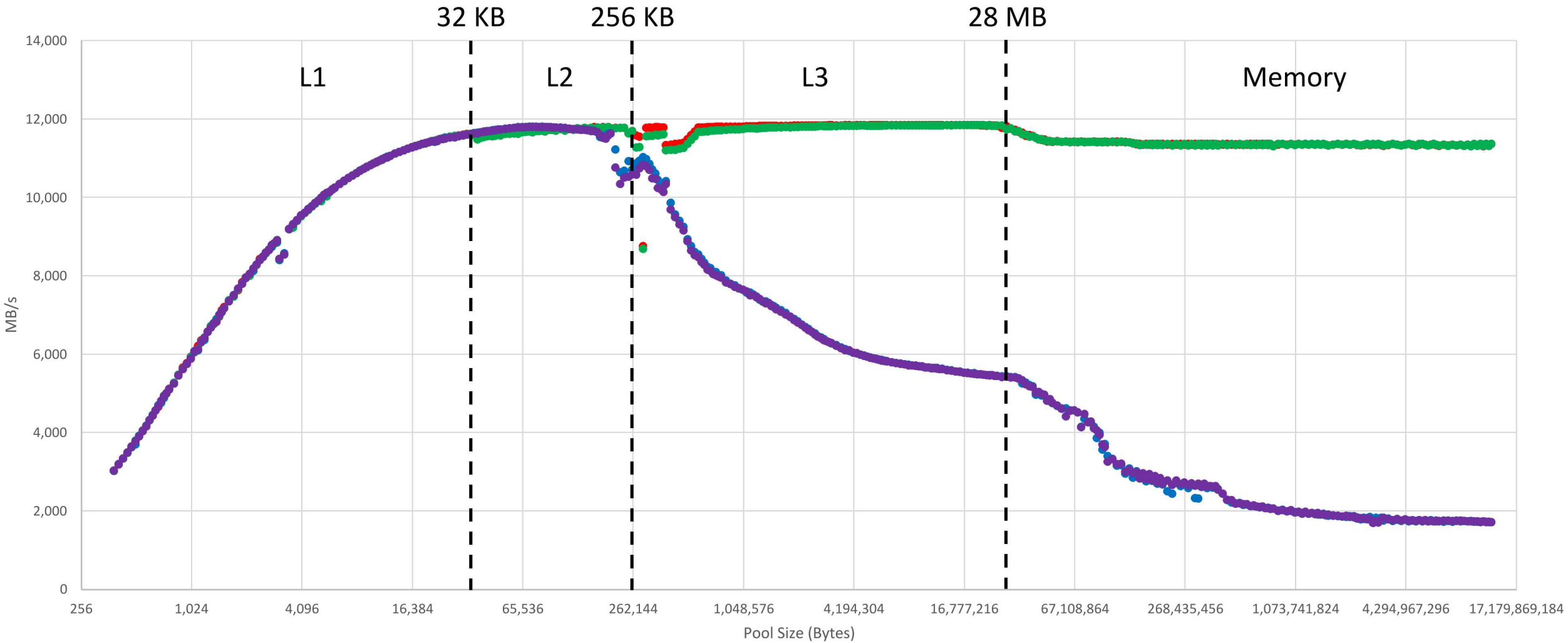
`y[i] = x[v[i]]`

`x`, `y` are double-precision floating-point vectors

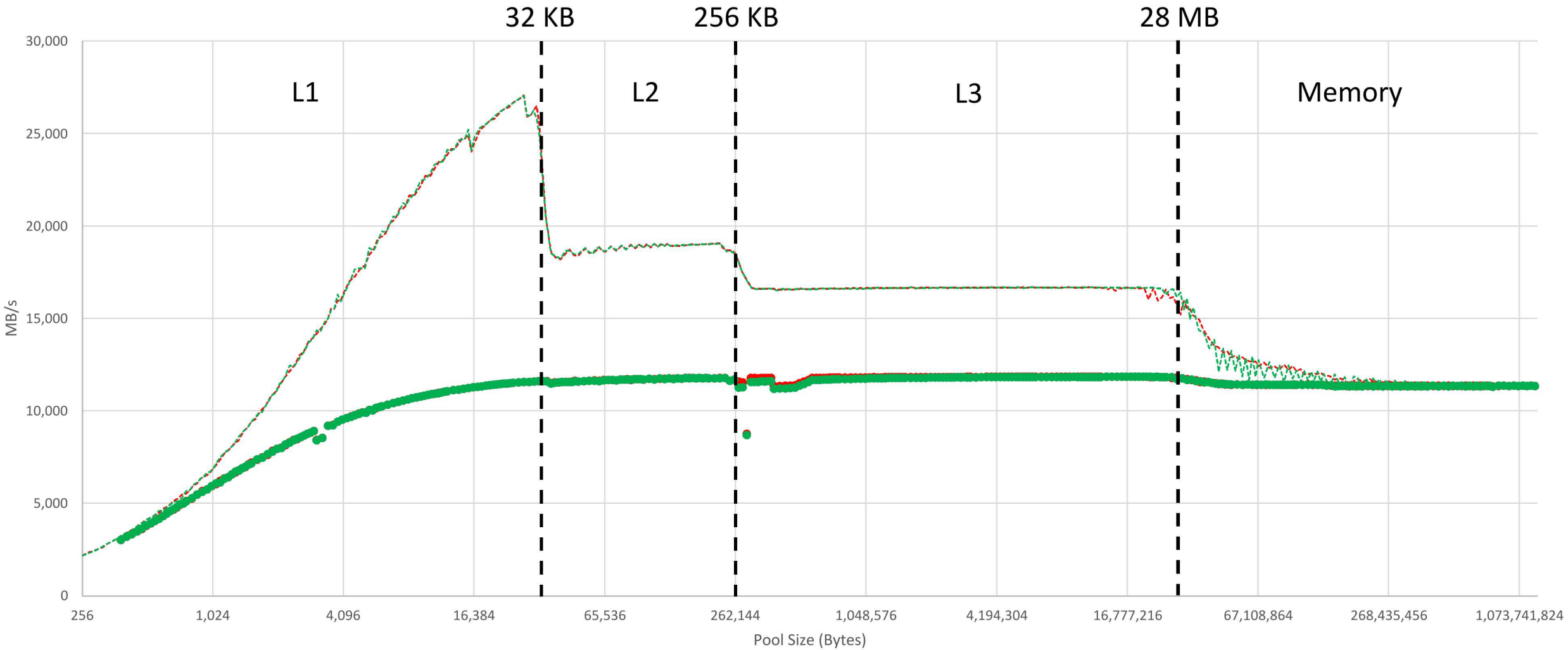
`v` is a 64-bit integer vector index into `x`

`i` is a 64-bit integer index

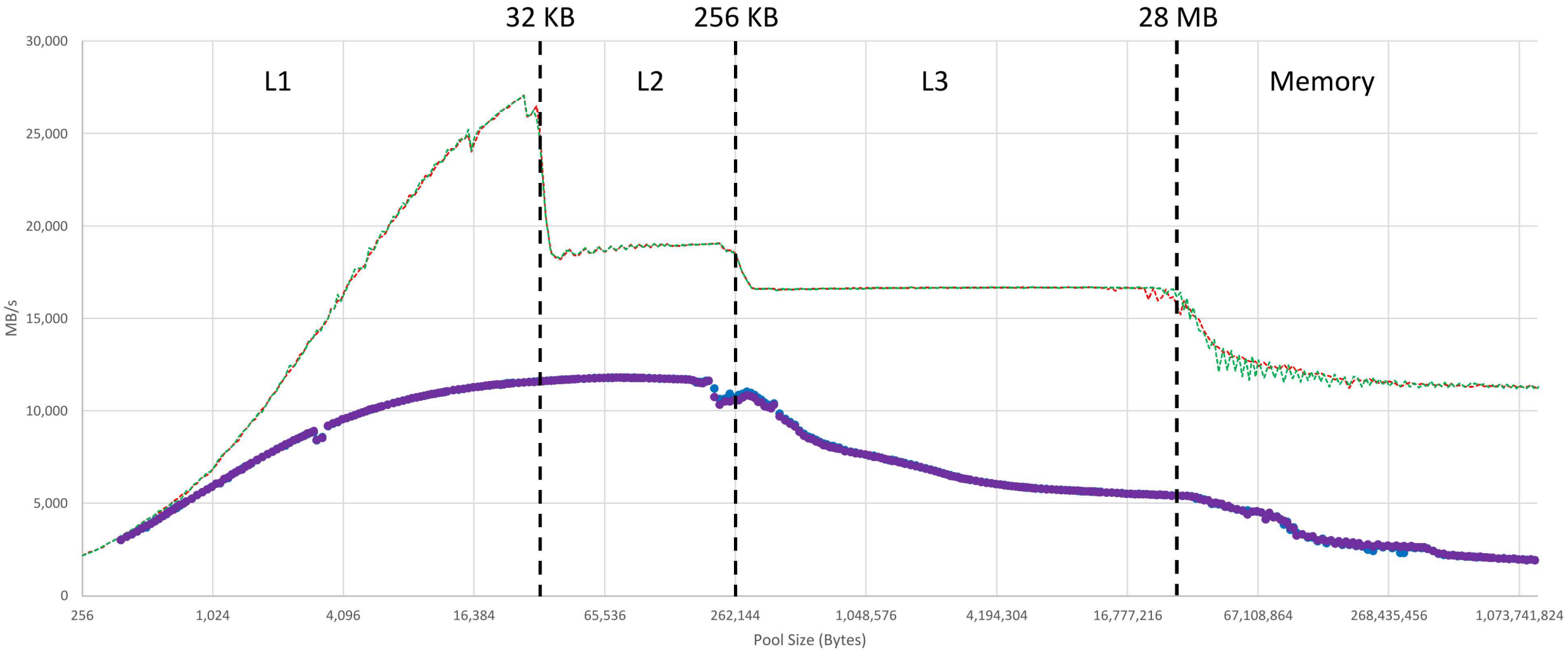
Gather MB/s (One Core)



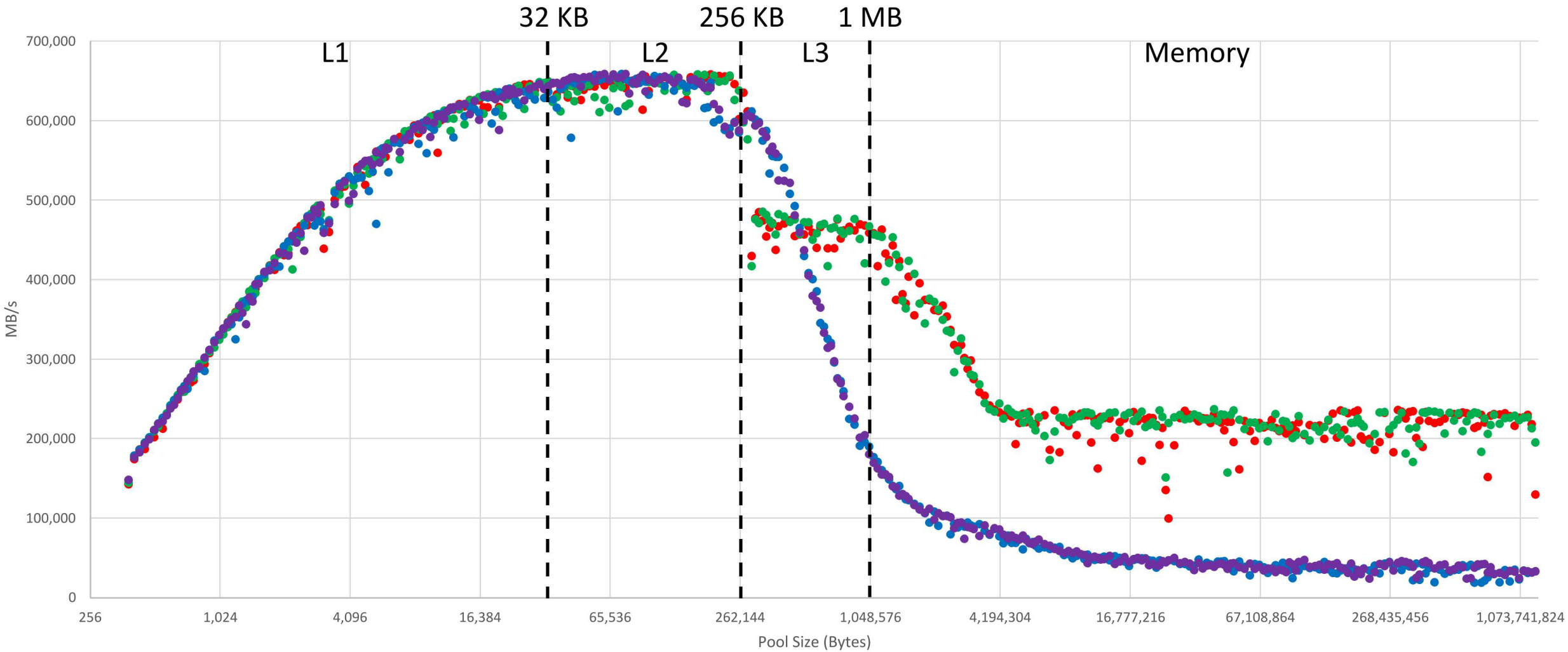
Sequential Gather MB/s (One Core)



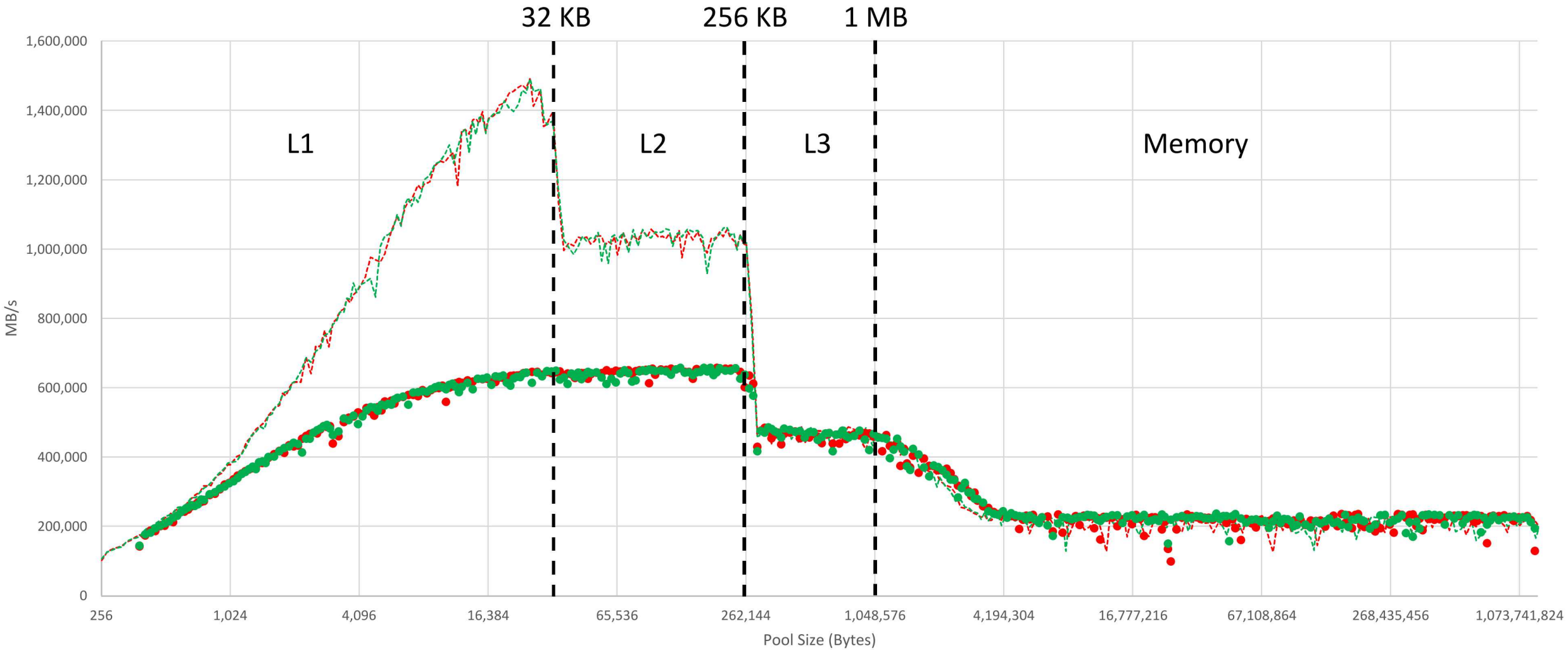
Random Gather MB/s (One Core)



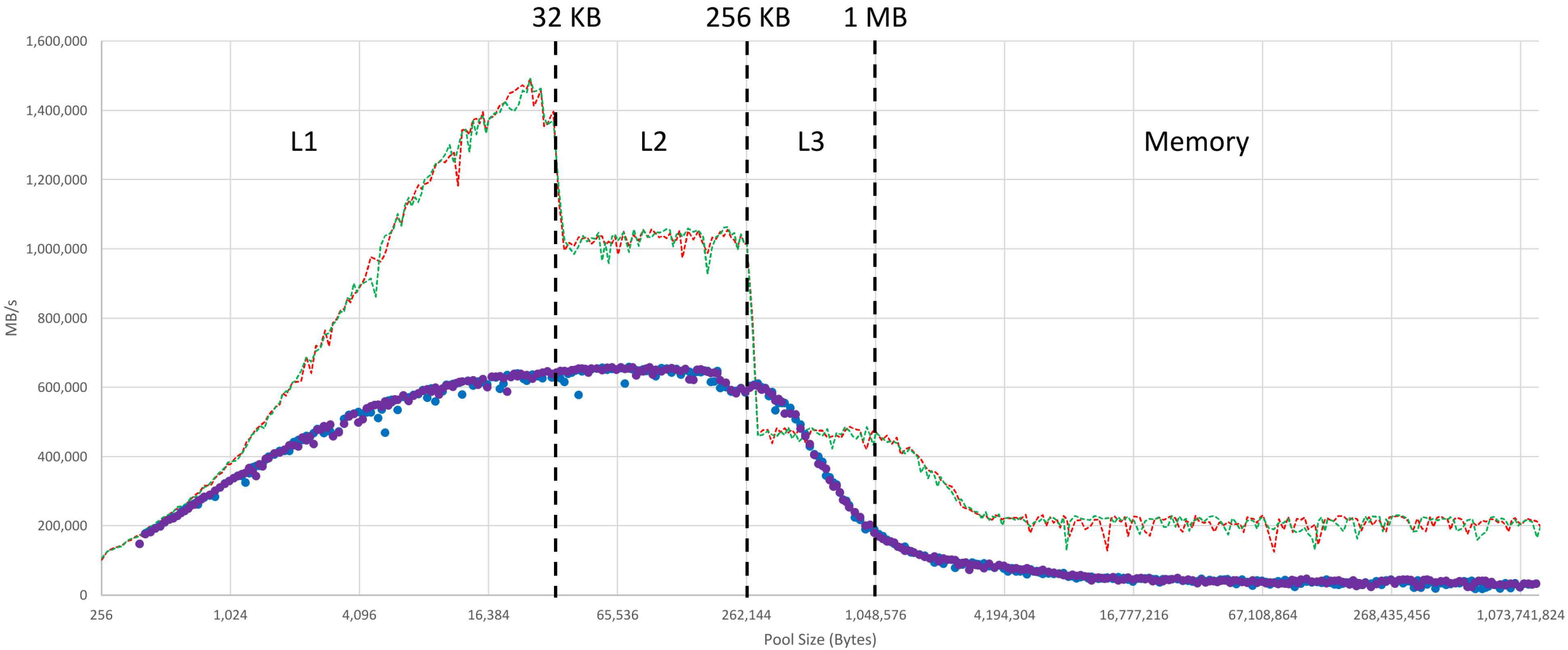
Gather MB/s (56 Cores)



Sequential Gather MB/s (56 Cores)



Random Gather MB/s (56 Cores)



Scatter

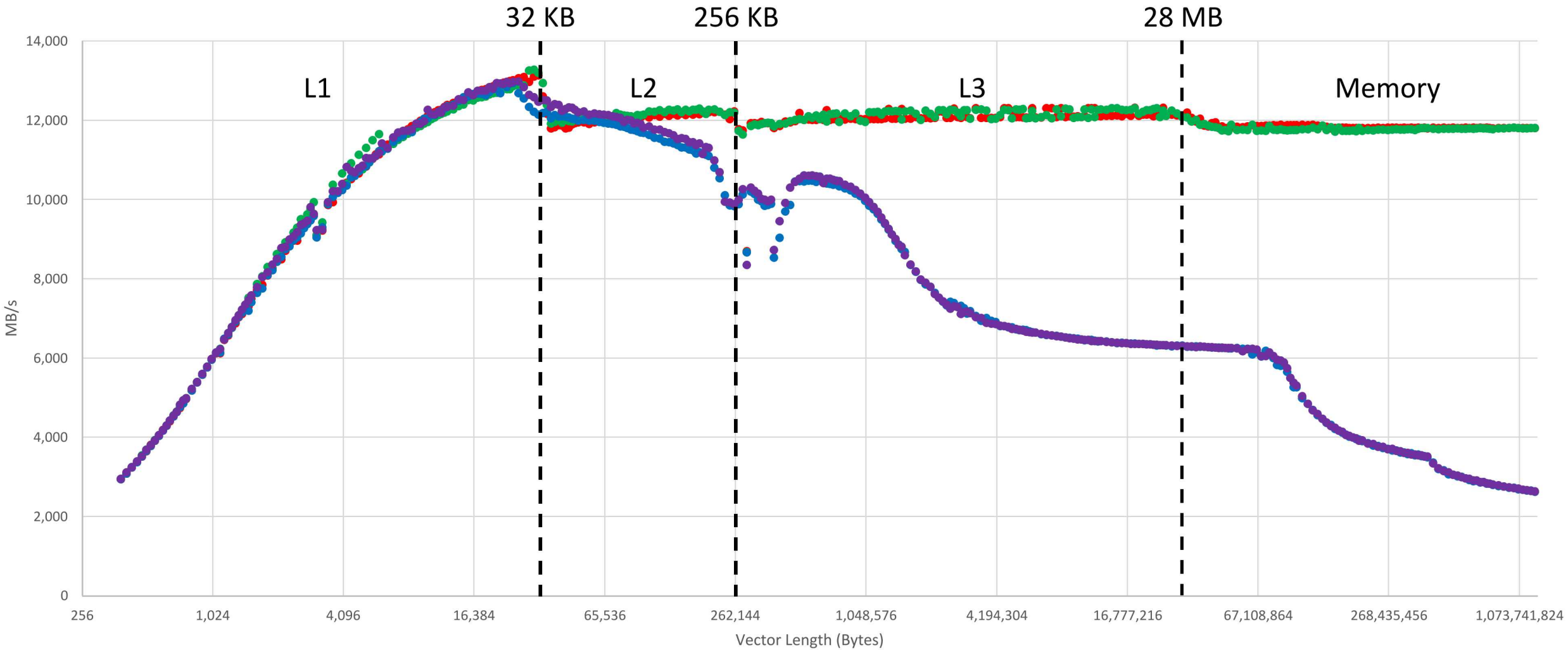
`y[v[i]] = x[i]`

`x`, `y` are double-precision floating-point vectors

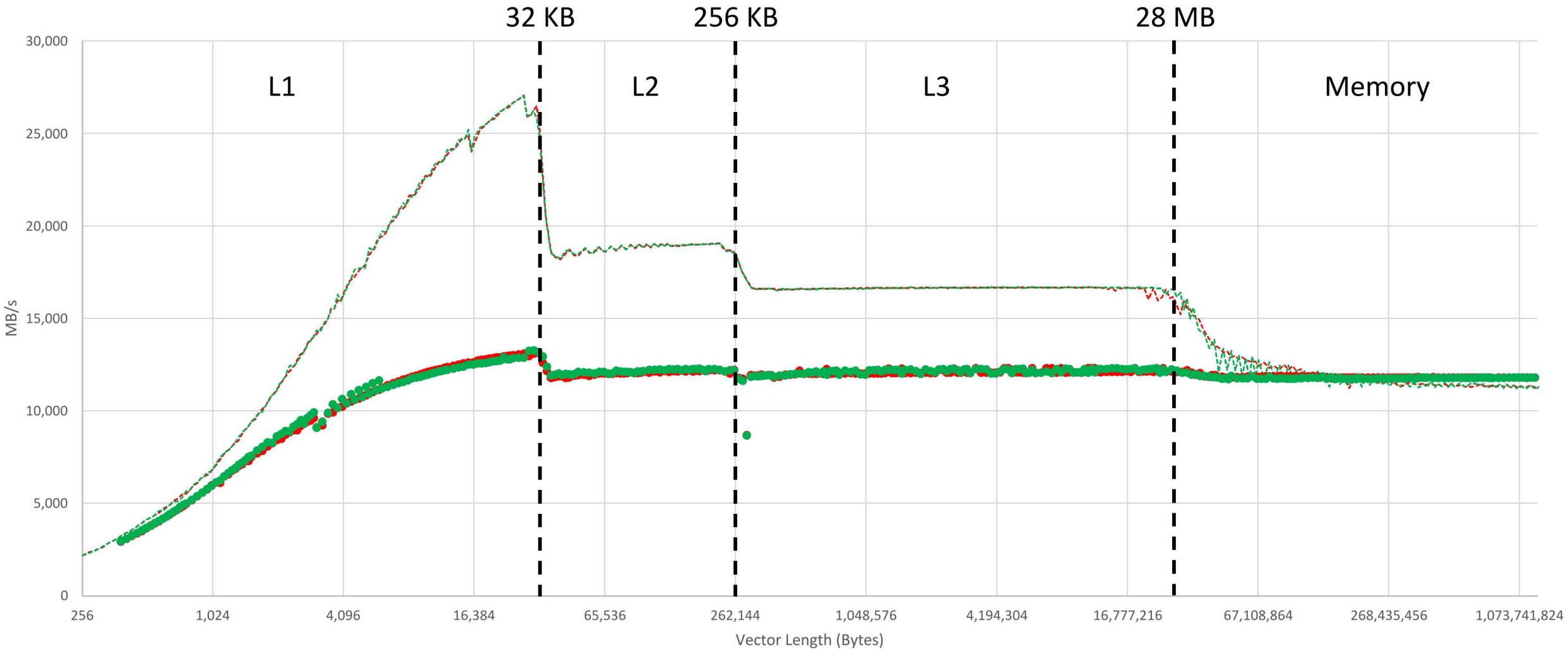
`v` is a 64-bit integer vector index into `x`

`i` is a 64-bit integer index

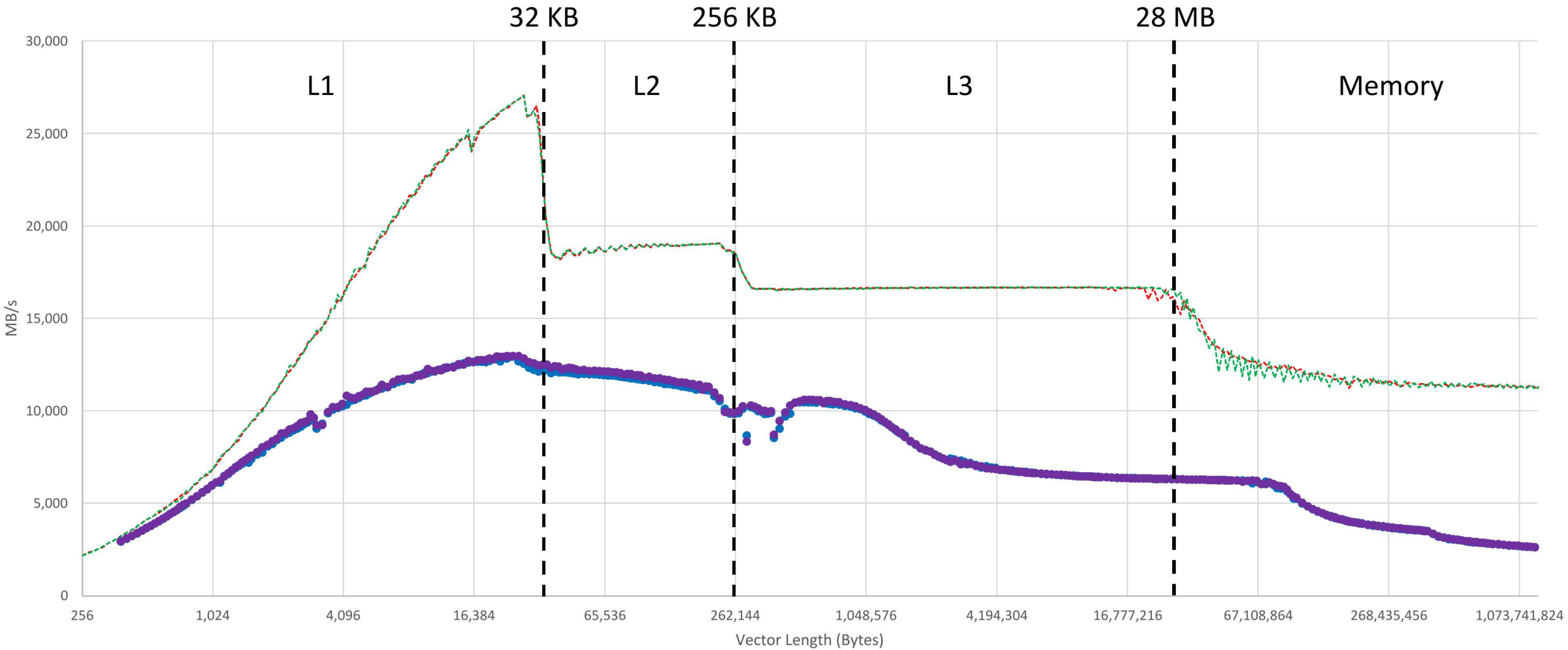
Scatter MB/s (One Core)



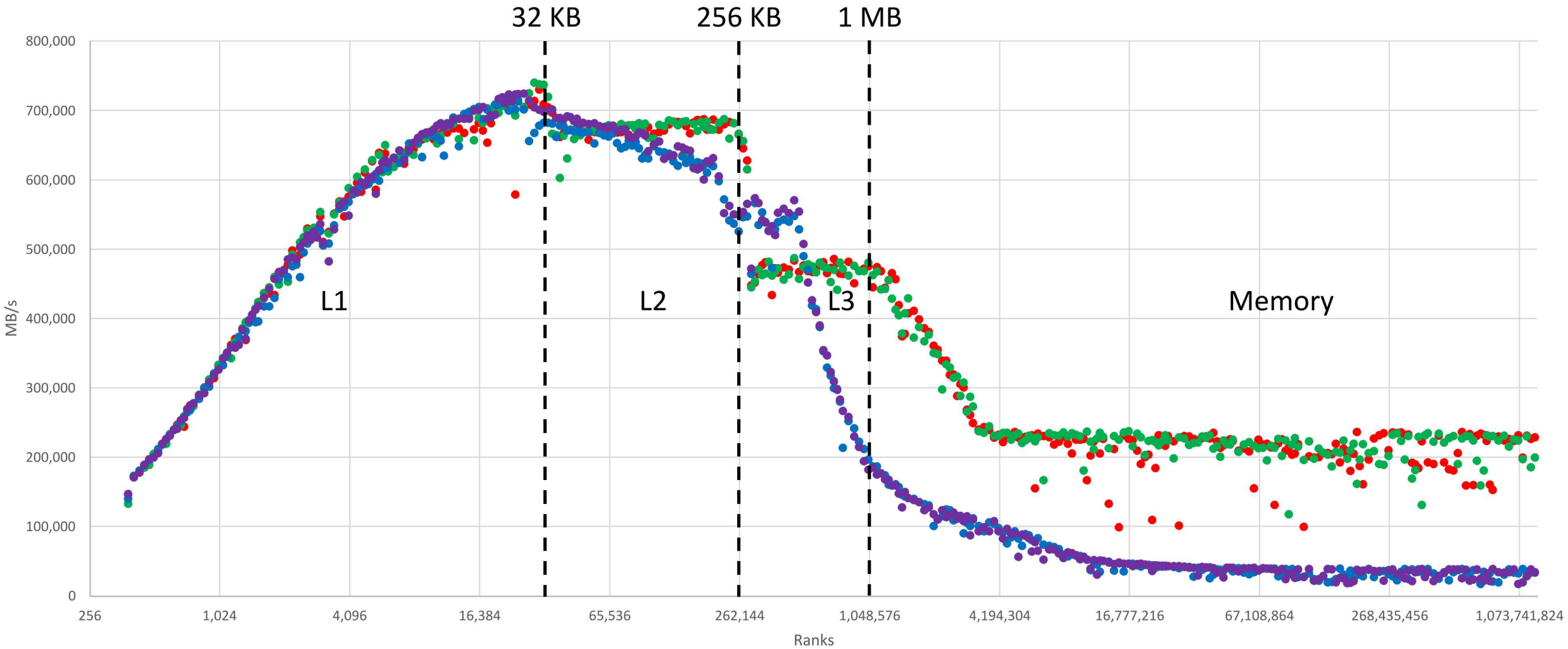
Sequential Scatter MB/s (One Core)



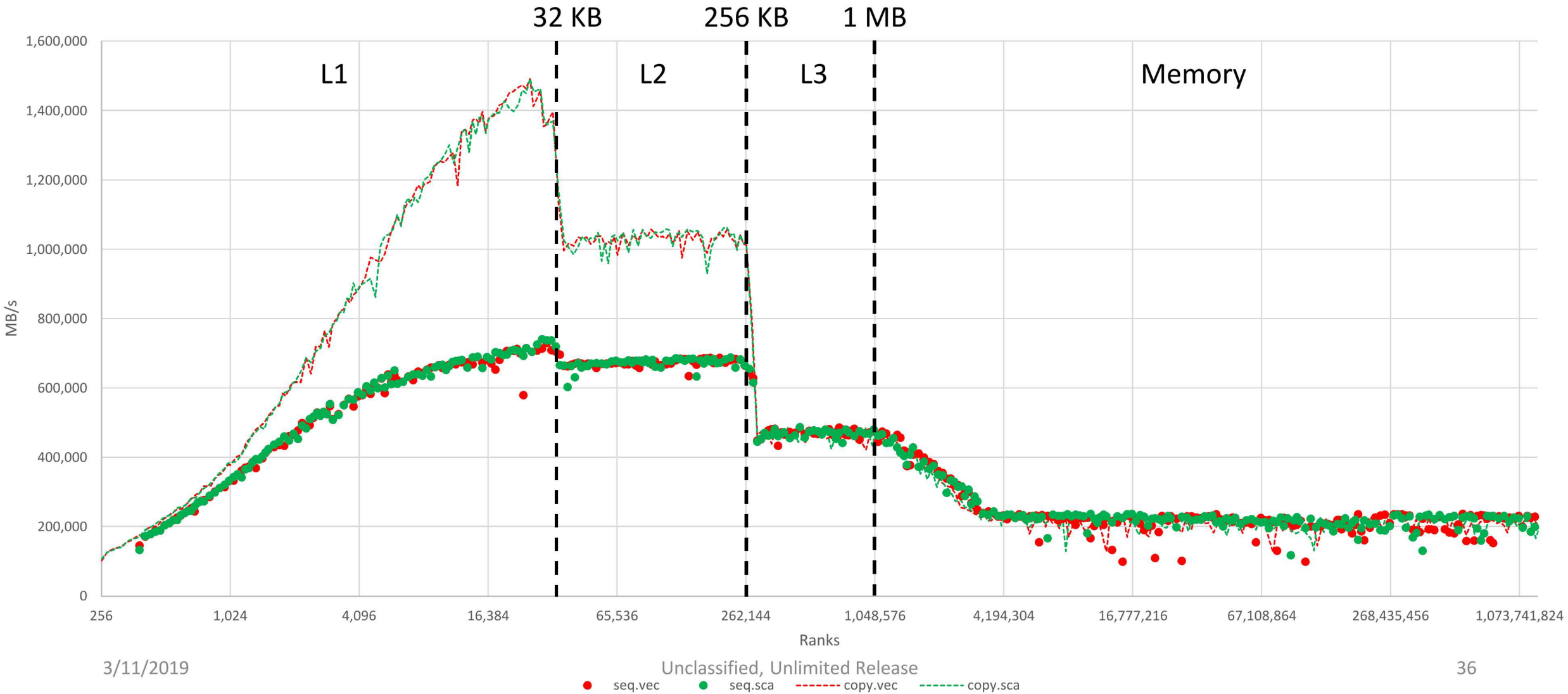
Random Scatter MB/s (One Core)



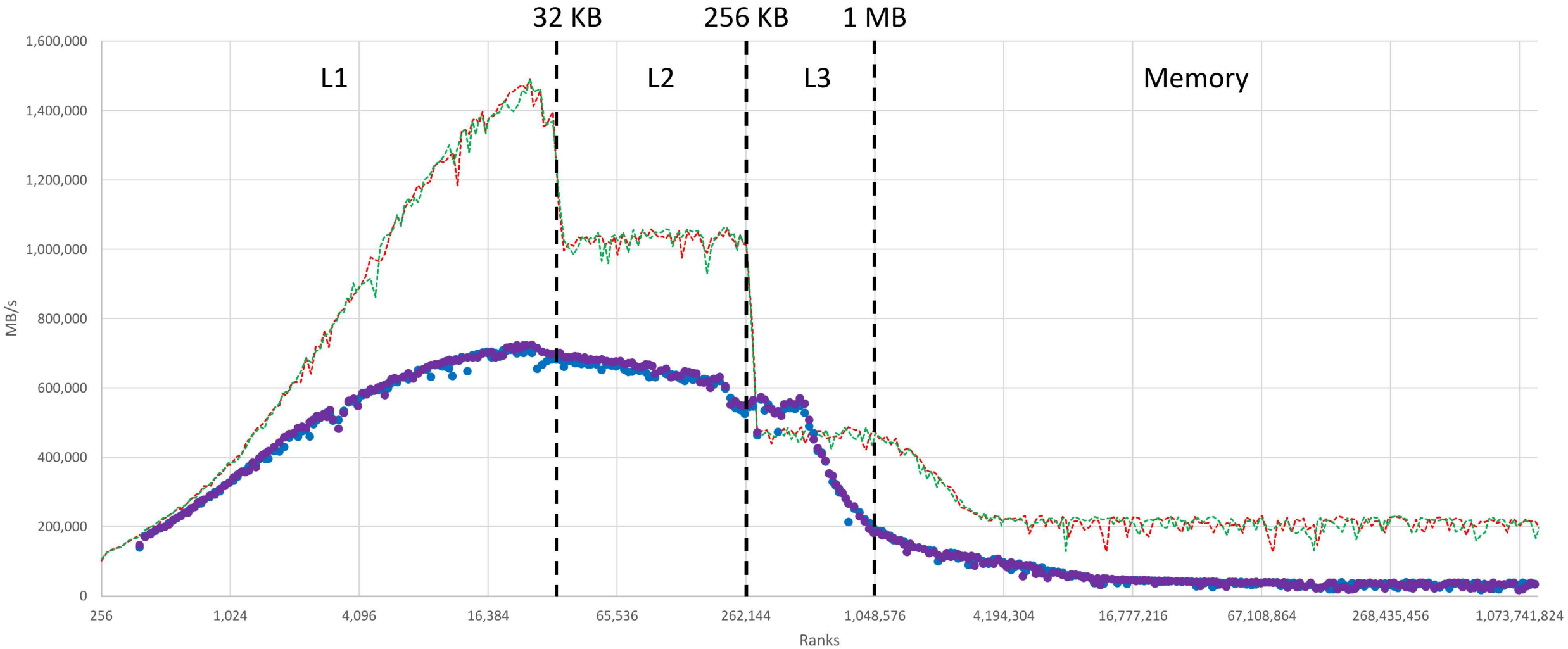
Scatter MB/s (56 Cores)



Sequential Scatter MB/s (56 Cores)

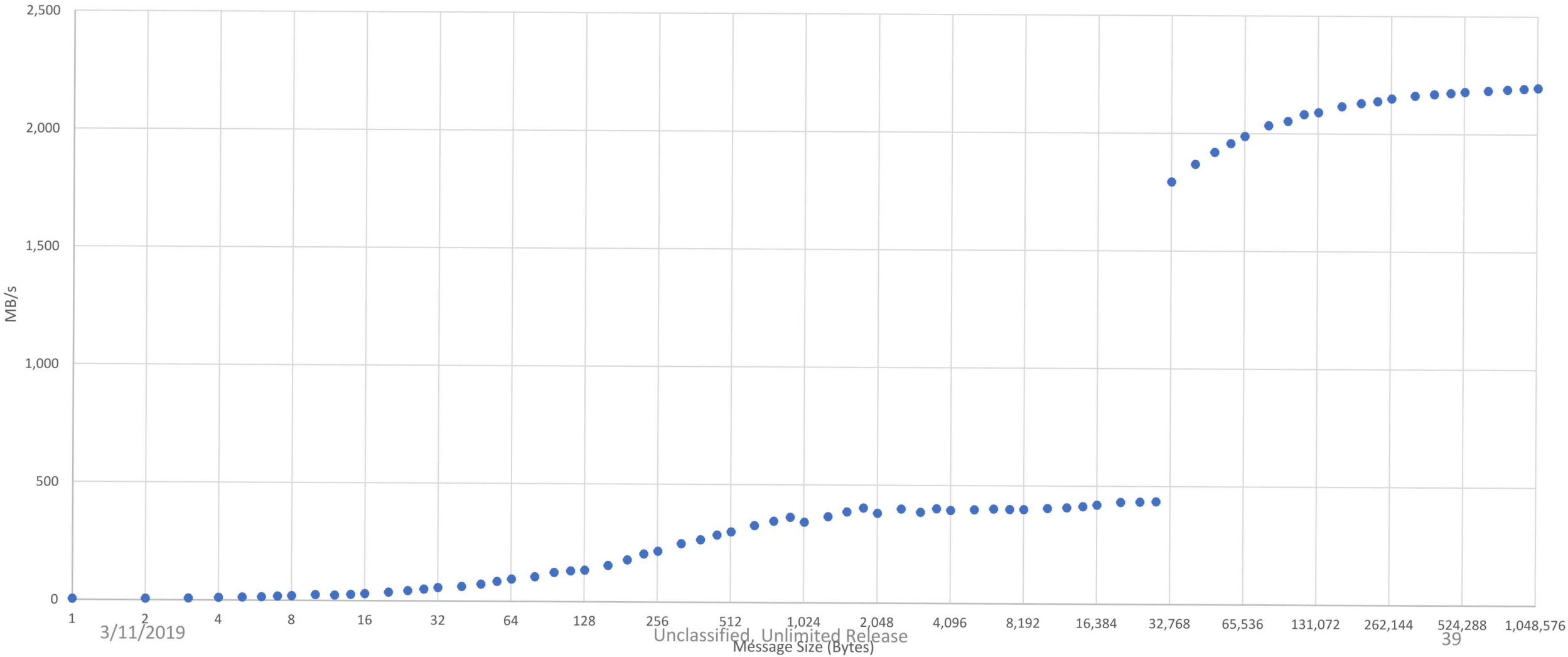


Random Scatter MB/s (56 Cores)

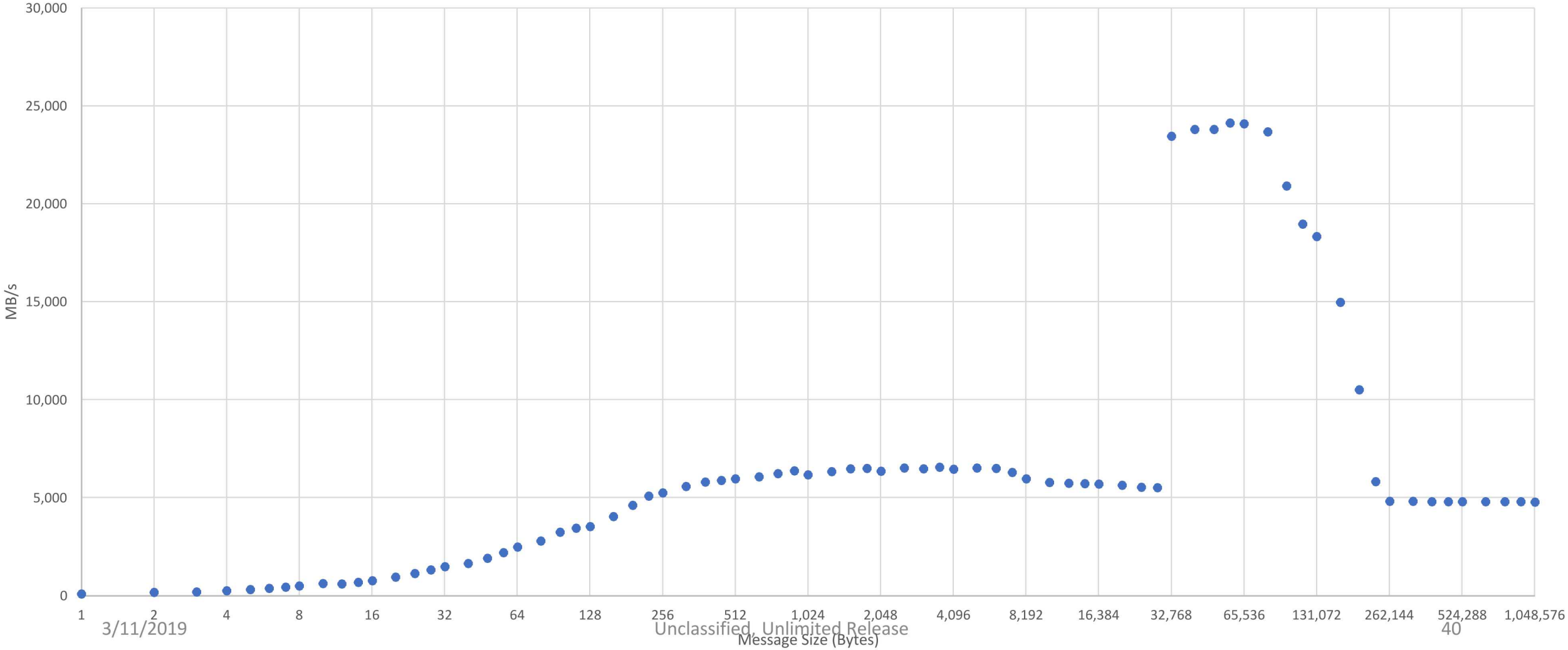


MPI_Sendrecv() & MPI_Allreduce()

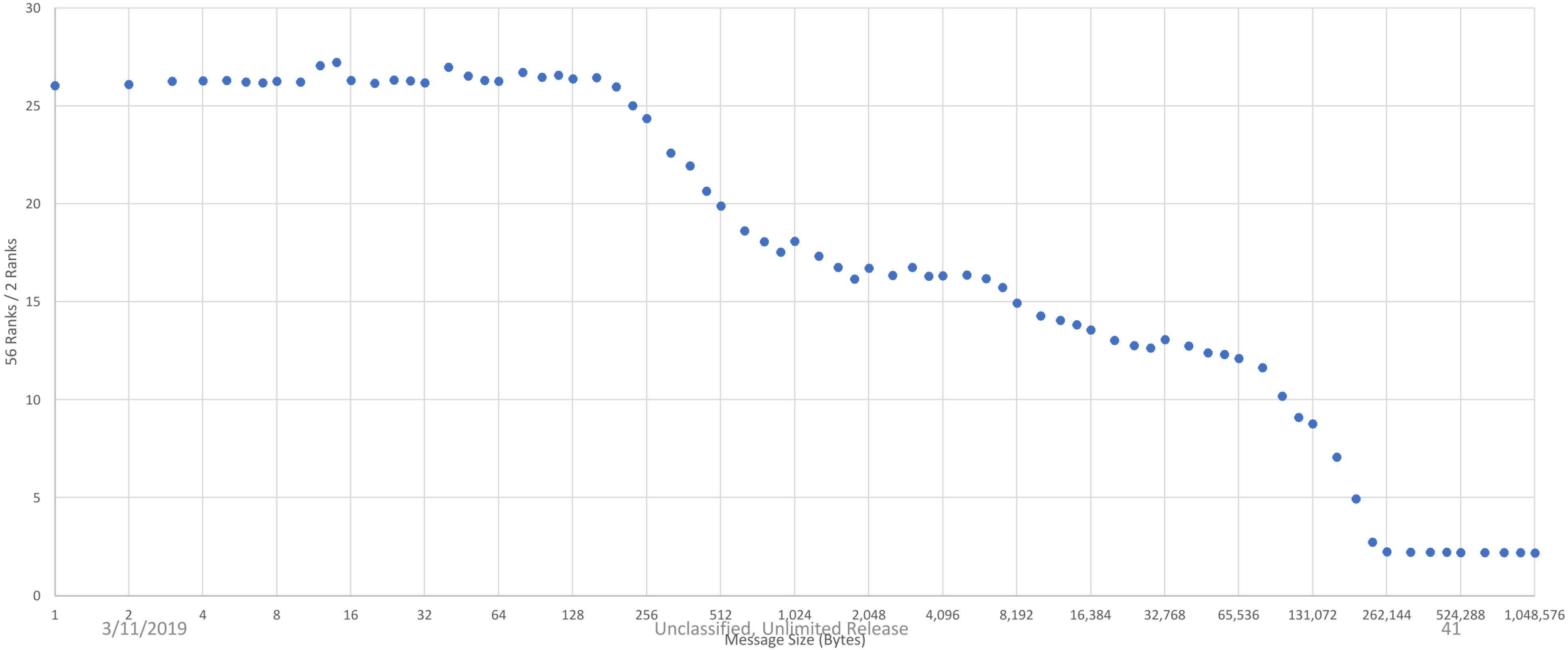
MPI_Sendrecv (1 Node, 2 Ranks)



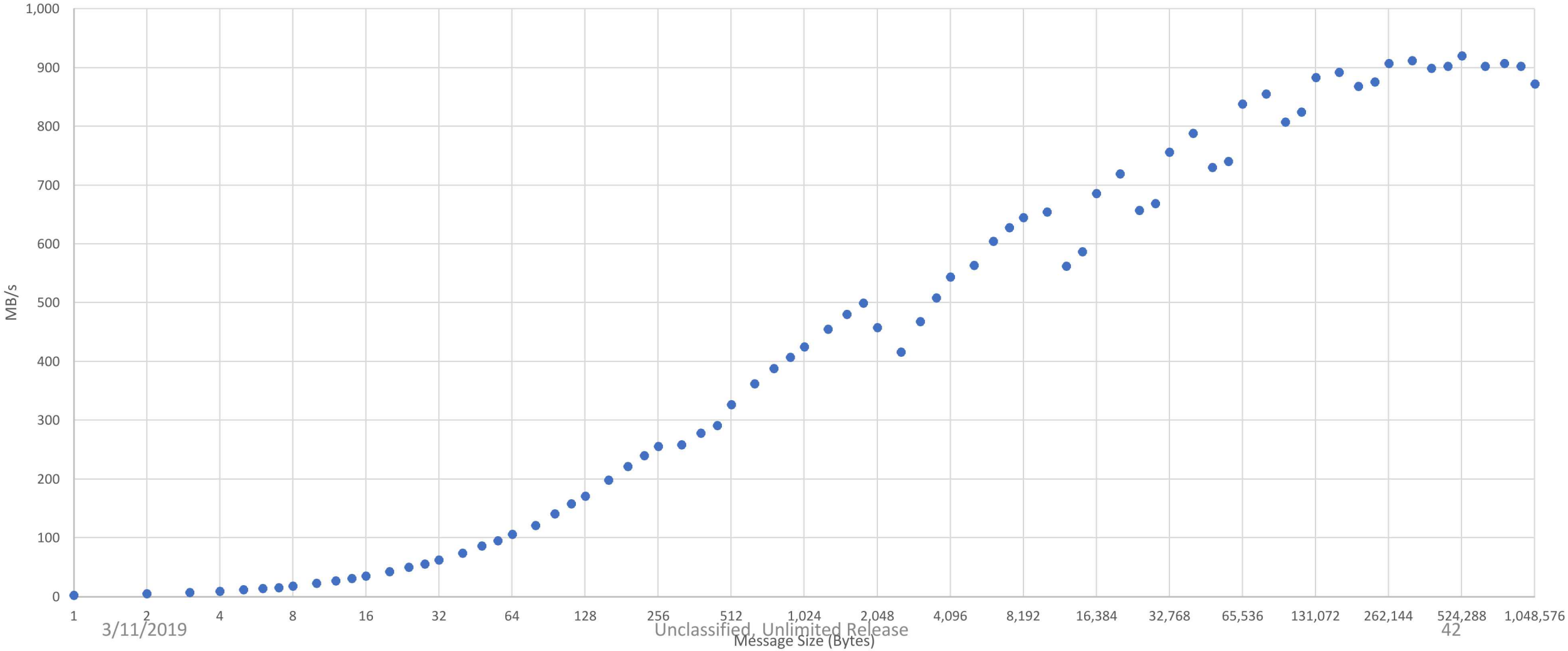
MPI_Sendrecv (1 Node, 56 Ranks)



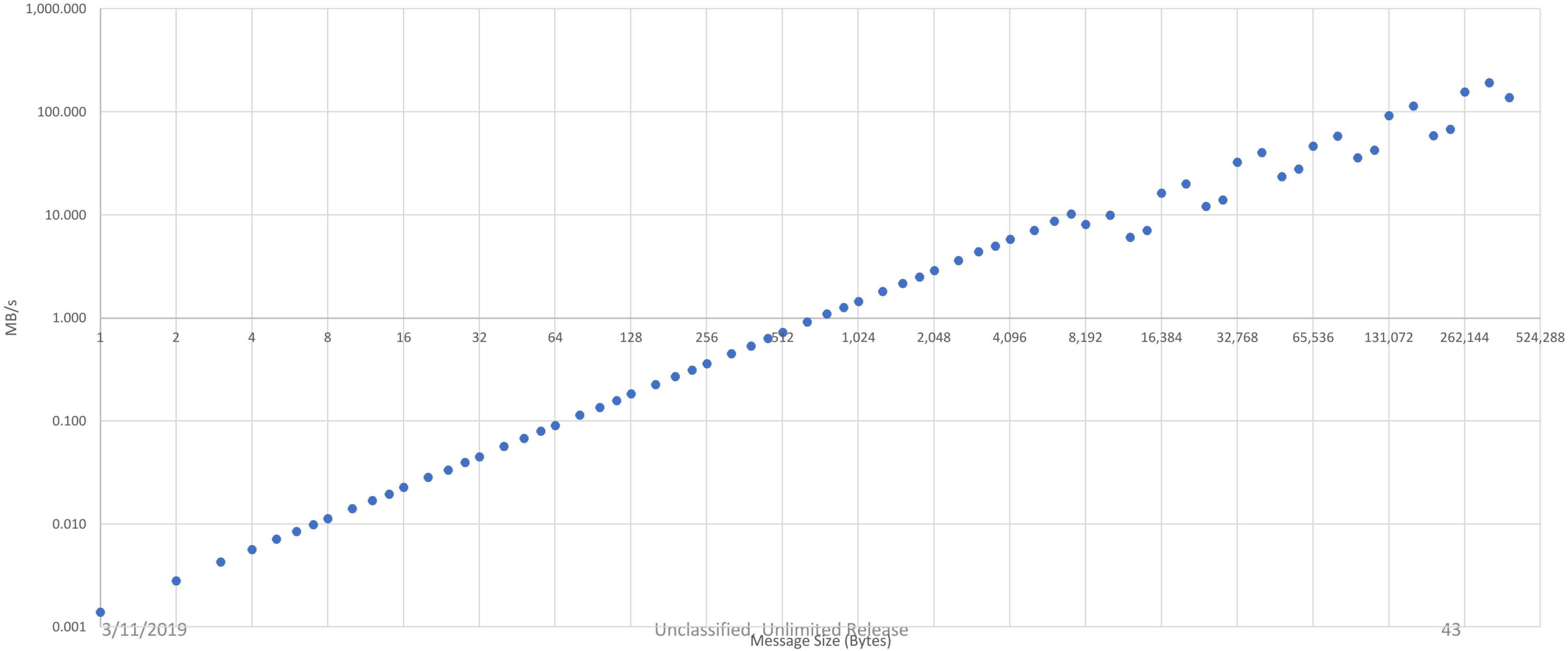
MPI_Sendrecv (56 Ranks / 2 Ranks)



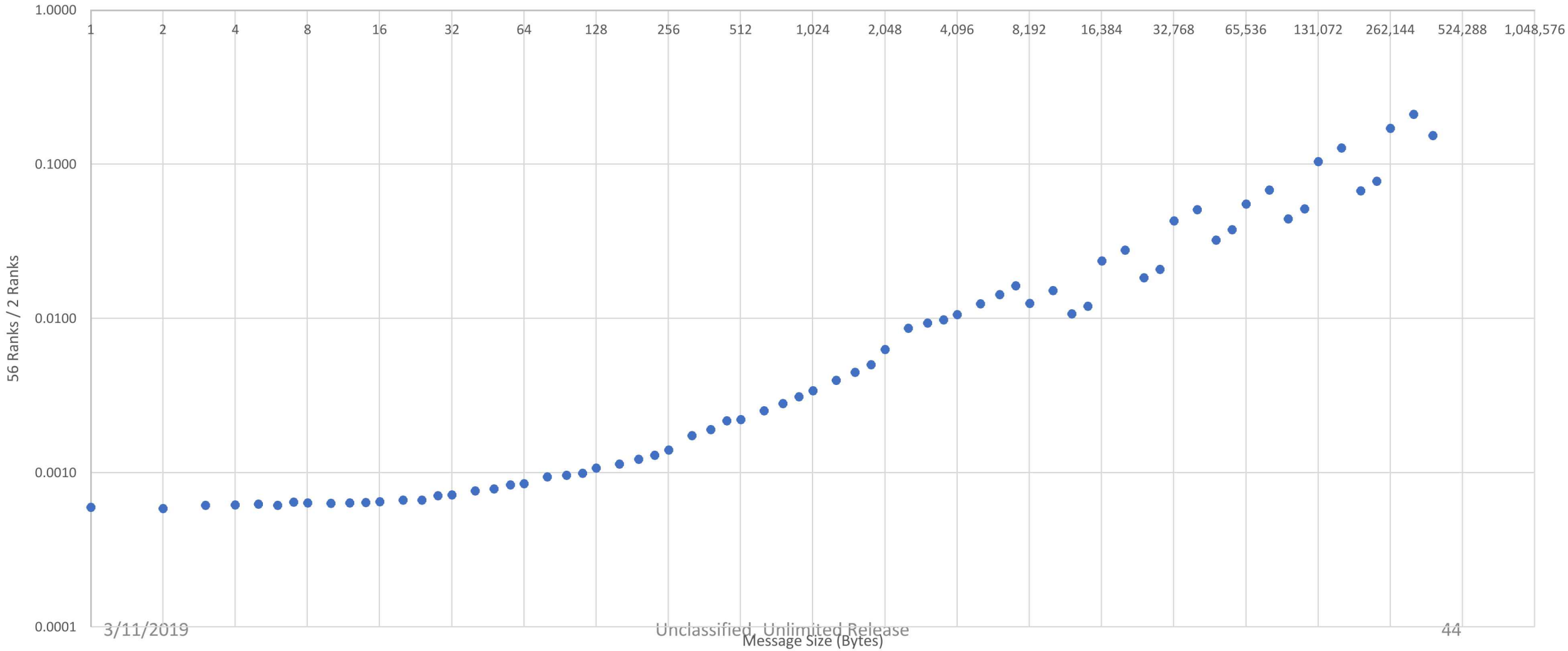
MPI_Allreduce (1 Node, 2 Ranks)



MPI_Allreduce (1 Node, 56 Ranks)



MPI_Allreduce (56 Ranks / 2 Ranks)



Questions?