MVAPICH
MPI, PGAS and Hybrid MPI+PGAS Library

HiBD
High-Performance Big Data

HiDL
High-Performance Deep Learning

SMART NIC BoF

BoF Session at SC '21 (Nov. '21)

by

Hari Subramoni

The Ohio State University

E-mail: subramon@cse.ohio-state.edu

https://web.cse.ohio-state.edu/~subramoni.1/

Follow us on

https://twitter.com/mvapich

Outline

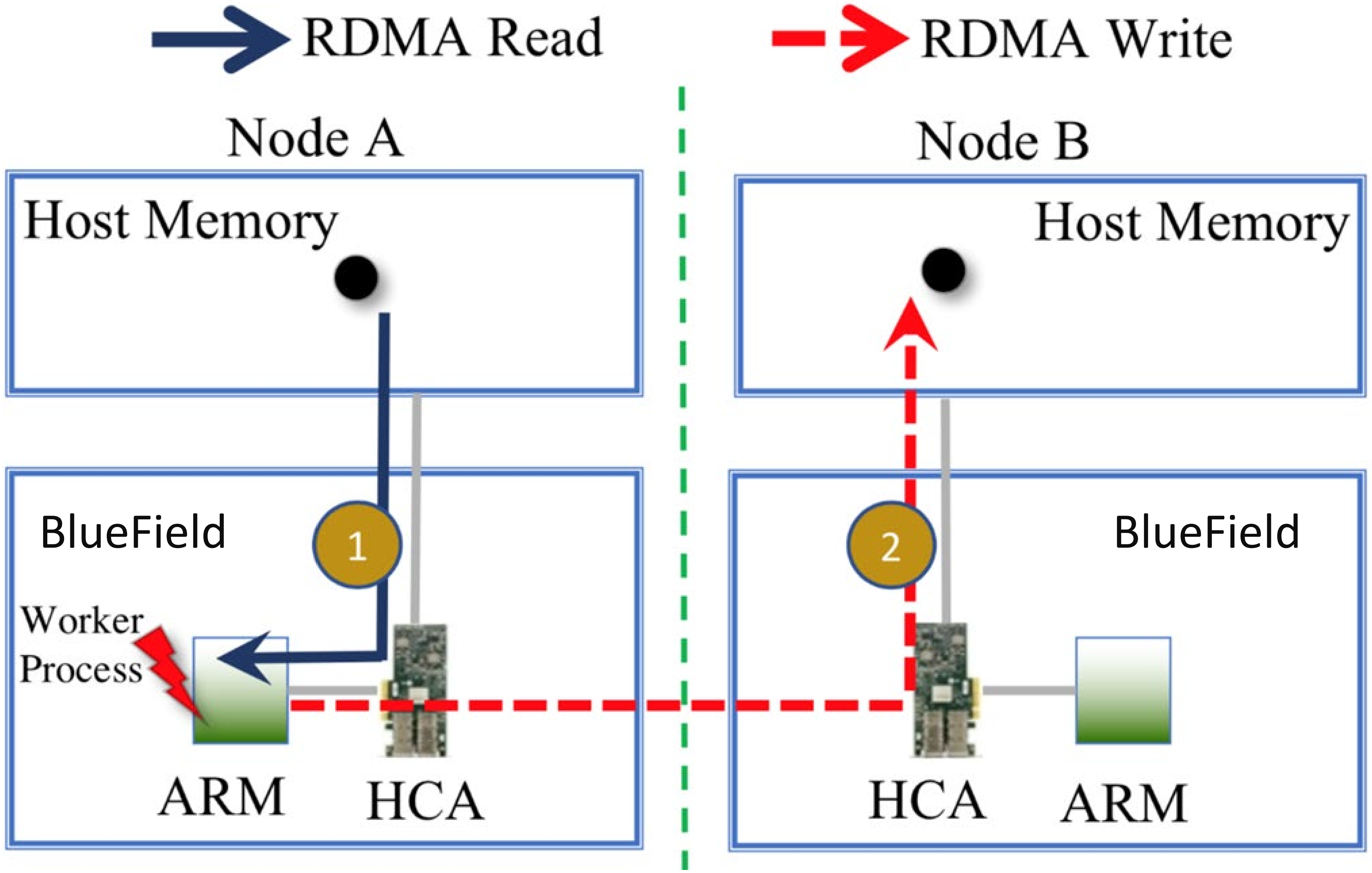
- Experience with SmartNICs
- Applications of SmartNICs
- Programming Models and Tools
- Architecture and Hardware

Proposed Offload Framework for SMART NICs

- Non-blocking collective operations are offloaded to a set of "worker processes"
- BlueField is set to separated host mode
- Worker processes are spawned to the ARM cores of BlueField
- Once the application calls a collective, host processes prepare a set of metadata and provide it to the Worker processes
- Using these metadata, worker processes can access host memory through RDMA
- Worker processes progress the collective on behalf of the host processes
- Once message exchanges are completed, worker processes notify the host processes about the completion of the non-blocking operation

Proposed Non-blocking Collective Designs

- Worker process performs RDMA Read to receive the data chunk from host main memory
- Once data is available in the ARM memory, worker process performs RDMA Write to the remote host memory



Overview of the MVAPICH2 Project

- High Performance open-source MPI Library
- Support for multiple interconnects
  - InfiniBand, Omni-Path, Ethernet/iWARP, RDMA over Converged Ethernet (RoCE), and AWS EFA
- Support for multiple platforms
  - x86, OpenPOWER, ARM, Xeon-Phi, GPGPUs (NVIDIA and AMD)
- Started in 2001, first open-source version demonstrated at SC ‘02
- Supports the latest MPI-3.1 standard
- http://mvapich.cse.ohio-state.edu
- Additional optimized versions for different systems/environments:
  - MVAPICH2-X (Advanced MPI + PGAS), since 2011
  - MVAPICH2-GDR with support for NVIDIA GPGPUs, since 2014
  - MVAPICH2-MIC with support for Intel Xeon-Phi, since 2014
  - MVAPICH2-Virt with virtualization support, since 2015
  - MVAPICH2-EA with support for Energy-Awareness, since 2015
  - MVAPICH2-Azure for Azure HPC IB instances, since 2019
  - MVAPICH2-X-AWS for AWS HPC+EFA instances, since 2019
- Tools:
  - OSU MPI Micro-Benchmarks (OMB), since 2003
  - OSU InfiniBand Network Analysis and Monitoring (INAM), since 2015

- Used by more than 3,200 organizations in 89 countries
- More than 1.52 Million downloads from the OSU site directly
- Empowering many TOP500 clusters (Nov. ‘21 ranking)
  - 4th , 10,649,600-core (Sunway TaihuLight) at NSC, Wuxi, China
  - 13th, 448, 448 cores (Frontera) at TACC
  - 26th, 288,288 cores (Lassen) at LLNL
  - 38th, 570,020 cores (Nurion) in South Korea and many others
- Available with software stacks of many vendors and Linux Distros (RedHat, SuSE, OpenHPC, and Spack)
- Partner in the 13th ranked TACC Frontera system
- Empowering Top500 systems for more than 16 years

Enhancing MVAPICH2 Software Architecture with DPU

High Performance Parallel Programming Models

- Message Passing Interface (MPI)
- PGAS (UPC, OpenSHMEM, CAF, UPC++)
- Hybrid --- MPI + X (MPI + PGAS + OpenMP/Cilk)

High Performance and Scalable Communication Runtime

Diverse APIs and Mechanisms

- Point-to-point Primitives
- Collectives Algorithms
- Job Startup
- Energy-Awareness
- Remote Memory Access
- I/O and File Systems
- Fault Tolerance
- Virtualization
- Active Messages
- Introspection & Analysis

Support for Modern Networking Technology

(InfiniBand, iWARP, RoCE, Omni-Path, Elastic Fabric Adapter)

- Transport Protocols: RC, XRC, UD, DC
- Modern Interconnect Features: UMR, ODP, SR-IOV, Multi Rail
- Modern HCA Features: Burst, Poll, Tag Match
- Modern IB Features: Multicast, SHARP, BlueField DPU

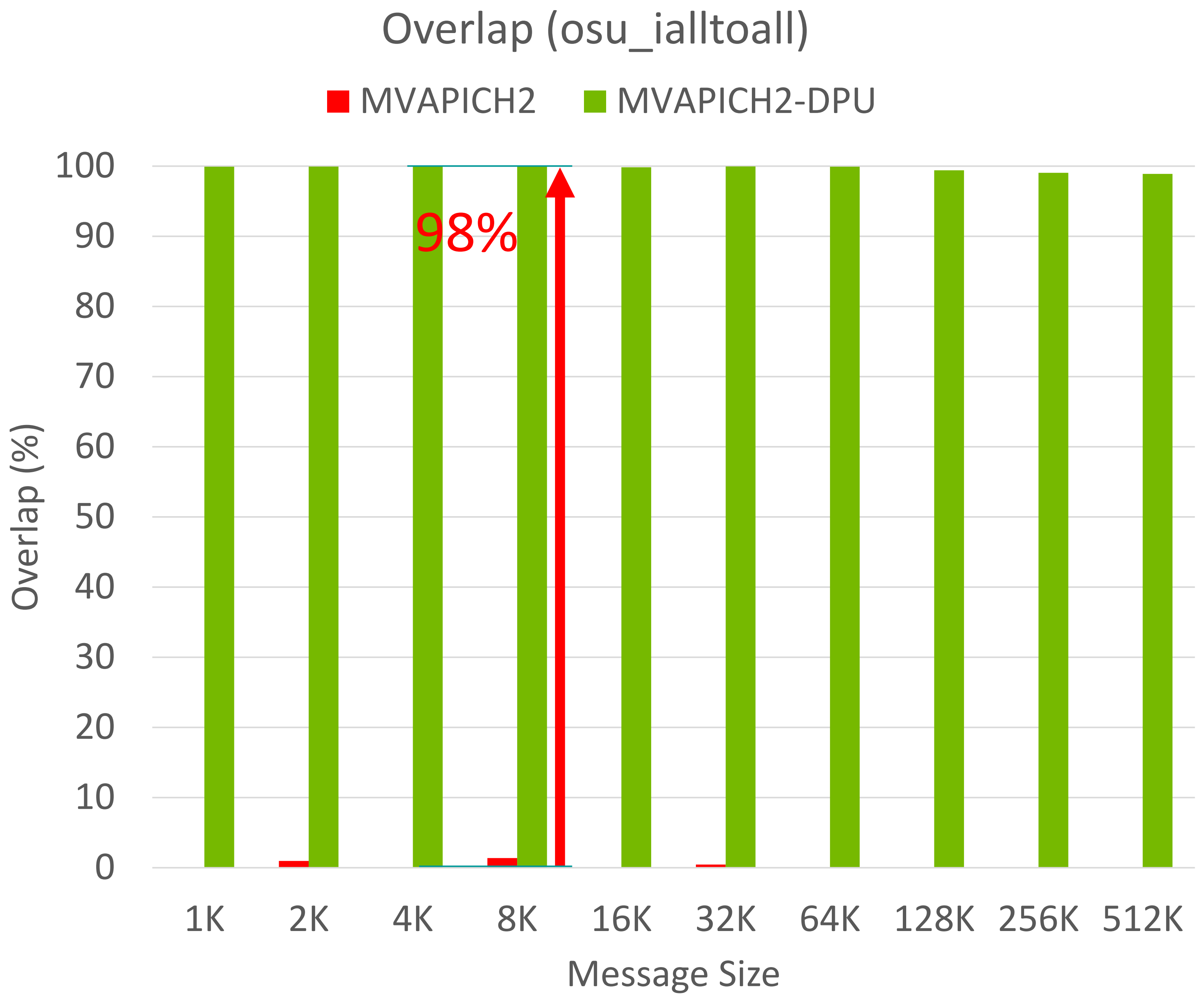
Experimental Setup for Performance Evaluation

- HPC Advisory Council High-Performance Computing Center
  - Cluster has 32 compute-node with Broadwell series of Xeon dual-socket, 16-core processors operating at 2.60 GHz with 128 GB RAM
  - NVIDIA BlueField-2 adapters are equipped with 8 ARM cores operating at 2.0 GHz with 16 GB RAM
- Based on the MVAPICH2-DPU MPI library
- OSU Micro Benchmark for nonblocking Alltoall and P3DFFT Application

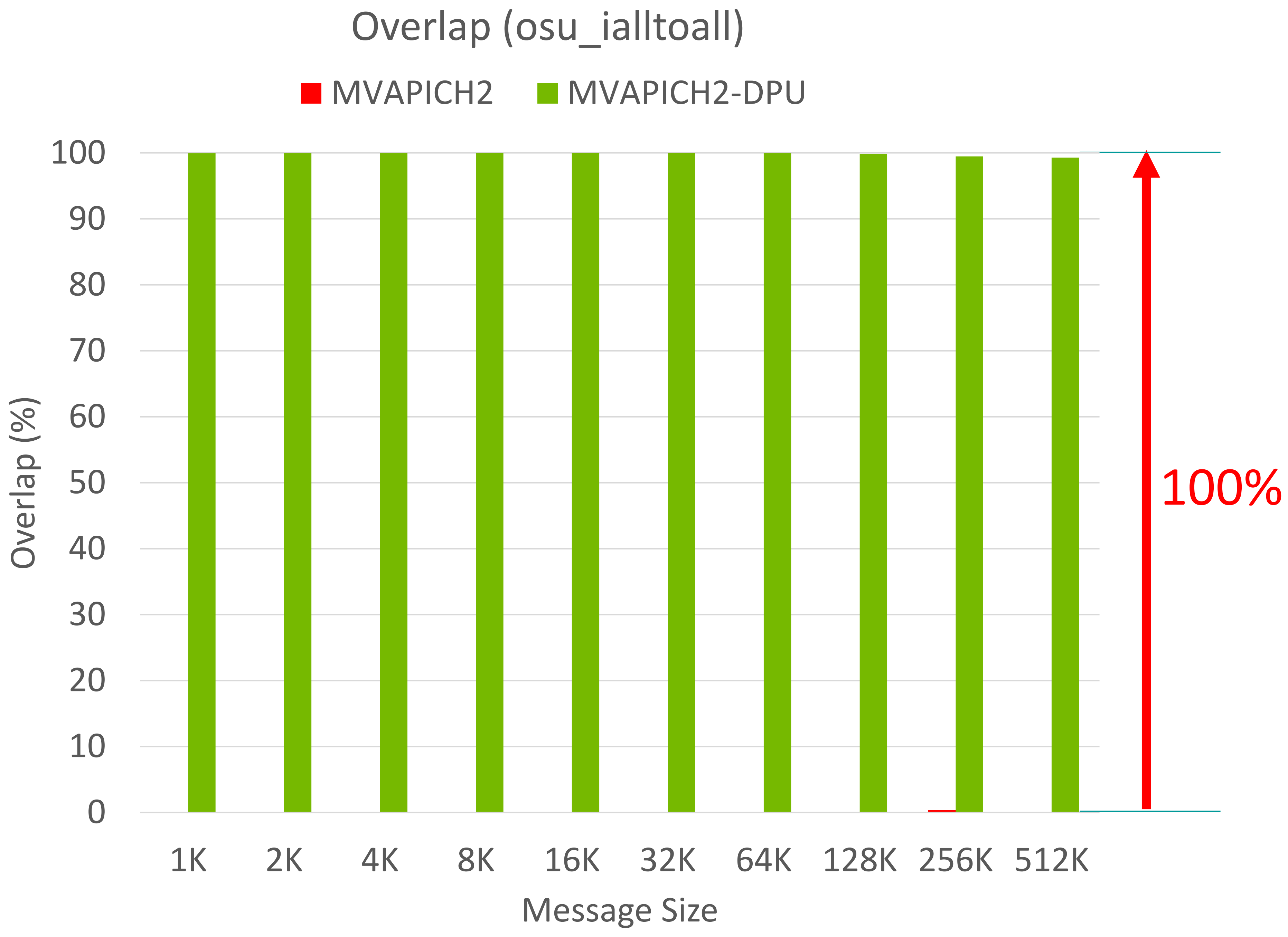
OSU Micro benchmark ialltoall

- `osu_ialltoall` benchmark metrics
  - Pure communication time
    - Latency t is measured by calling MPI_Ialltoall followed by MPI_Wait
  - Total execution time
    - Total T = MPI_Ialltoall + synthetic compute + MPI_Wait
  - Overlap
    - Benchmark creates a synthetic computation block that takes t microsecond to finish. Before starting compute, MPI_Ialltoall is called and after that MPI_Wait. Overlap is calculated based on total execution time and compute time.
  - Part of the standard OSU Micro-Benchmark

Overlap of Communication and Computation with osu_Ialltoall (32 nodes)



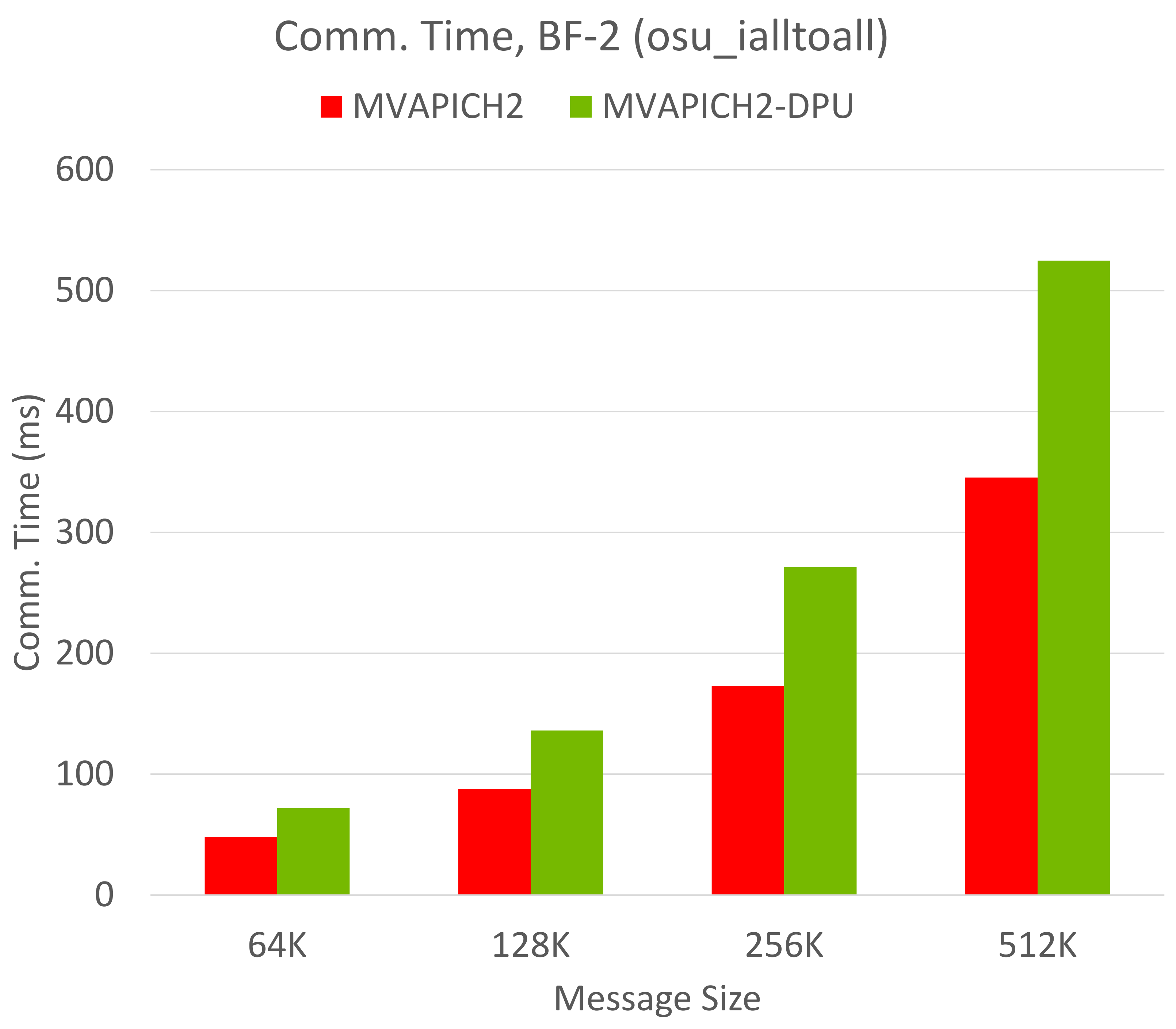
32 Nodes, 16 PPN



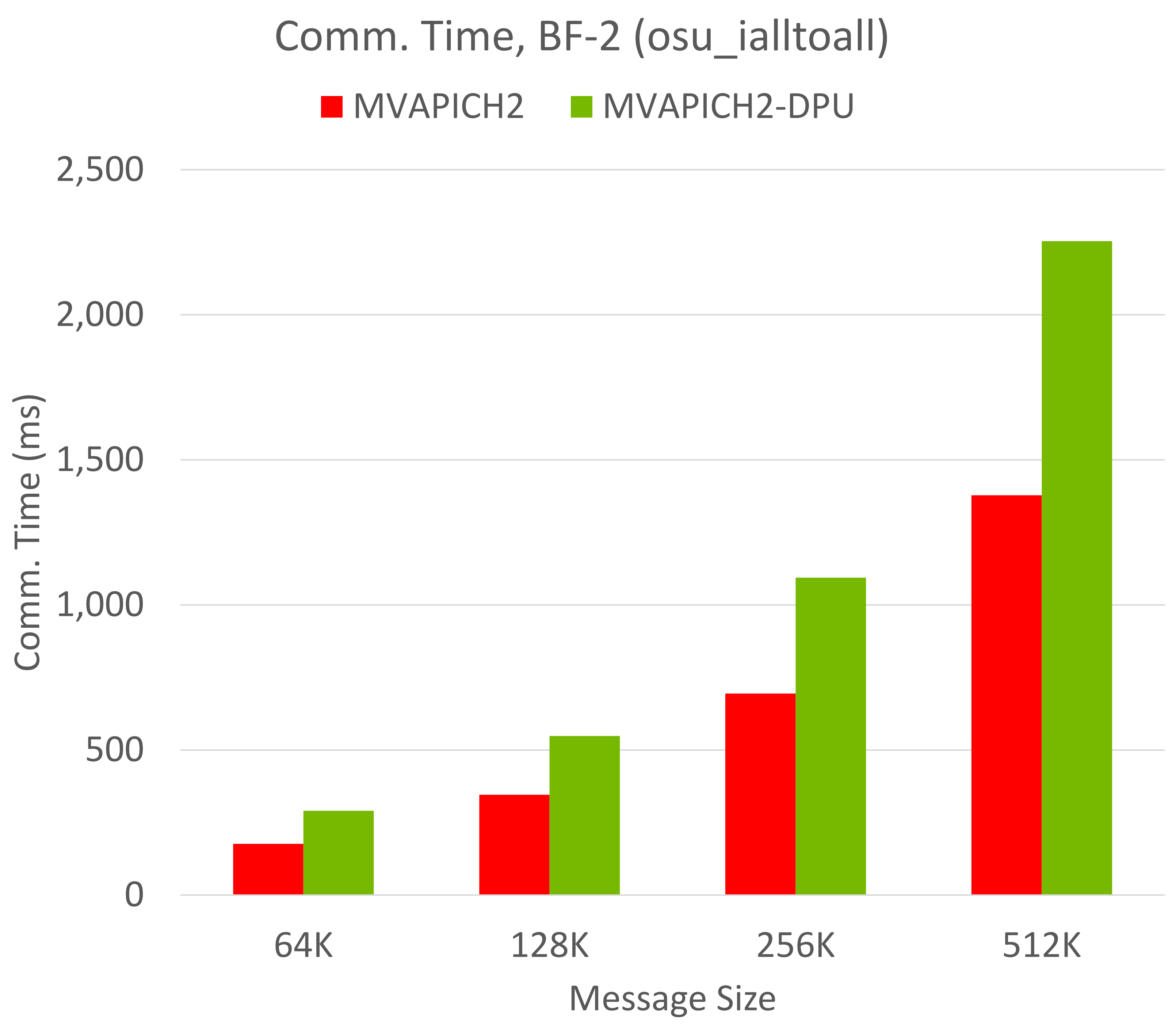
32 Nodes, 32 PPN

Delivers peak overlap

Pure Communication Latency with osu_ialltoall (32 nodes)

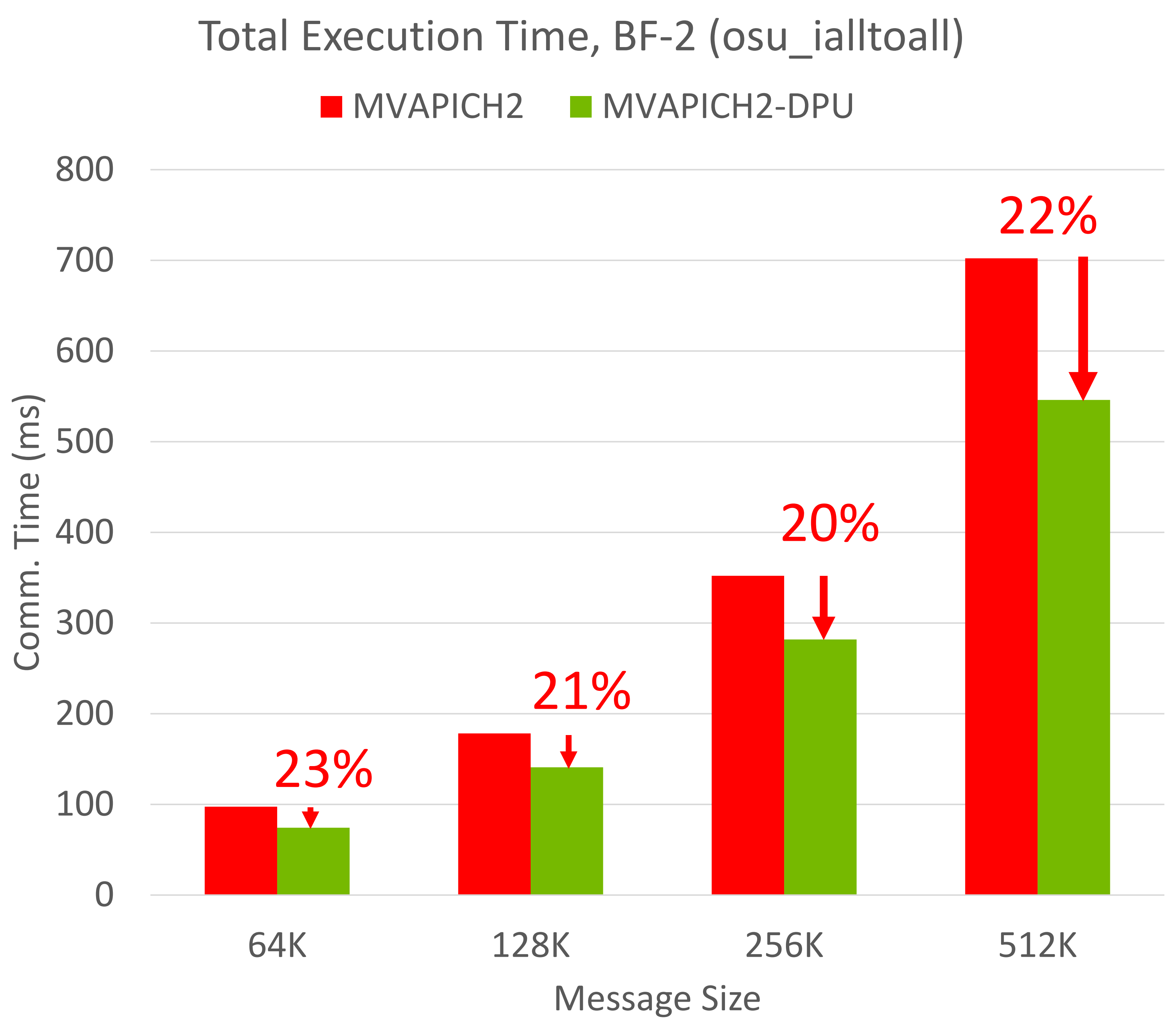


32 Nodes, 16 PPN

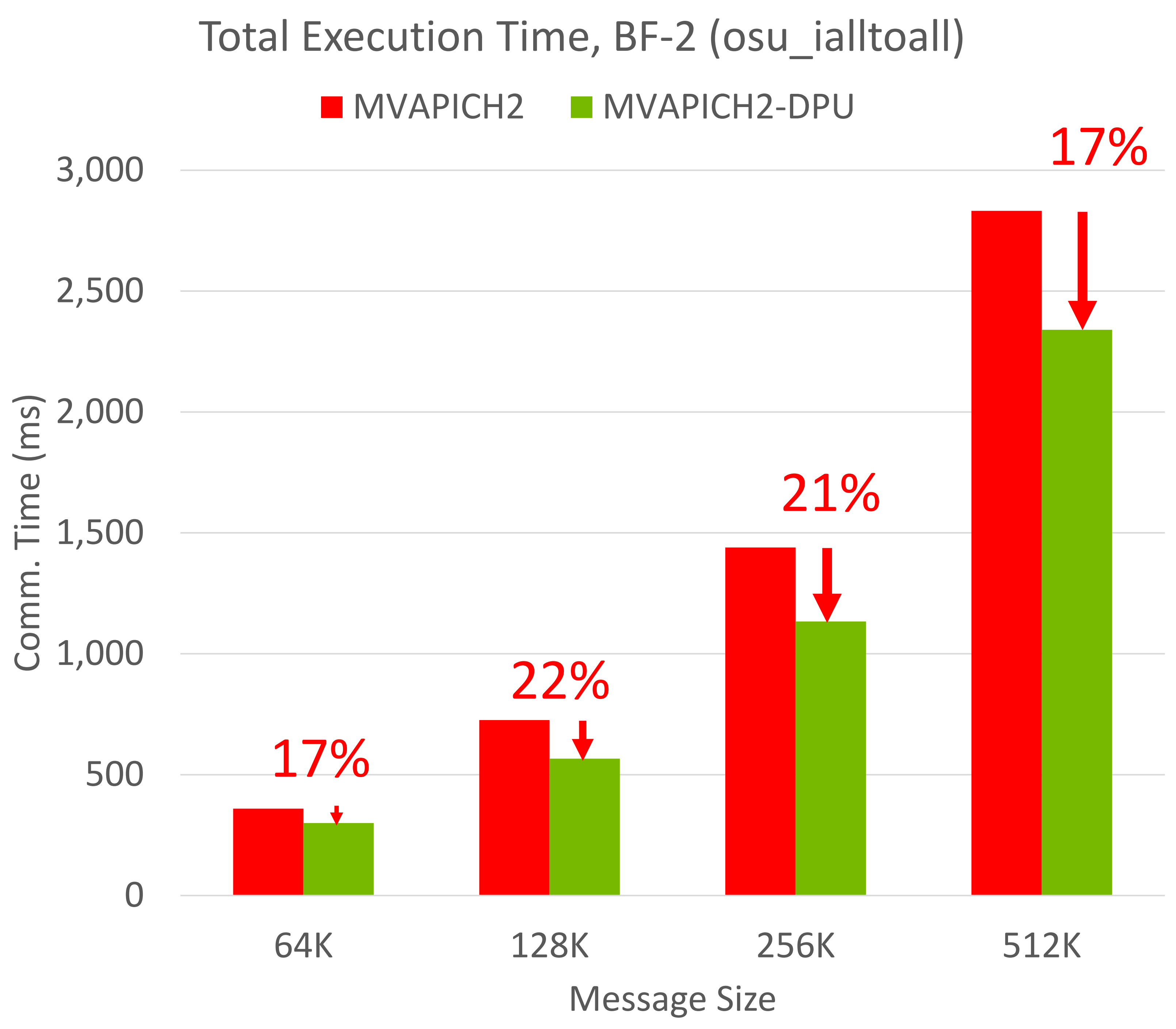


32 Nodes, 32 PPN

Total Execution Time with osu_ialltoall (32 nodes)



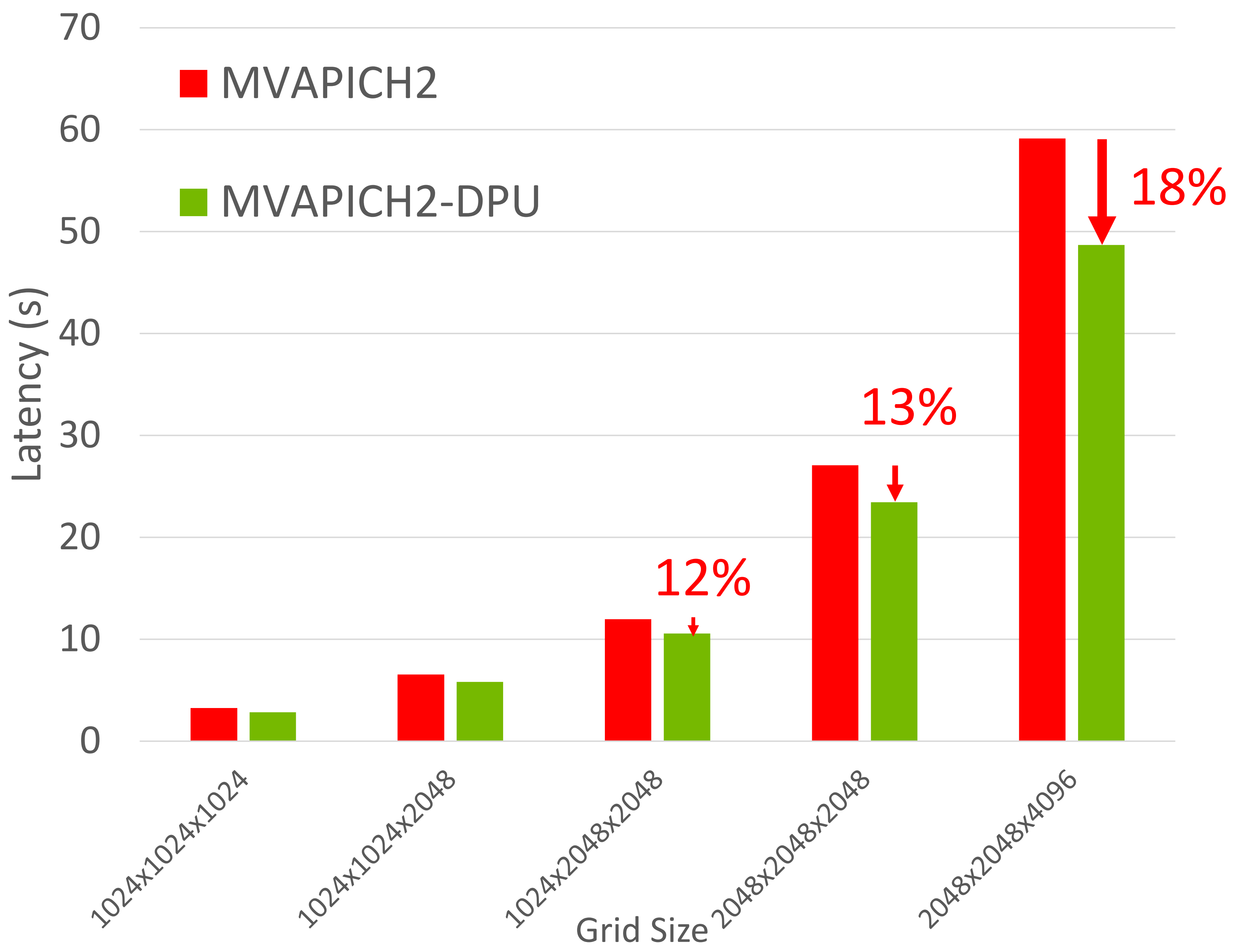
32 Nodes, 16 PPN



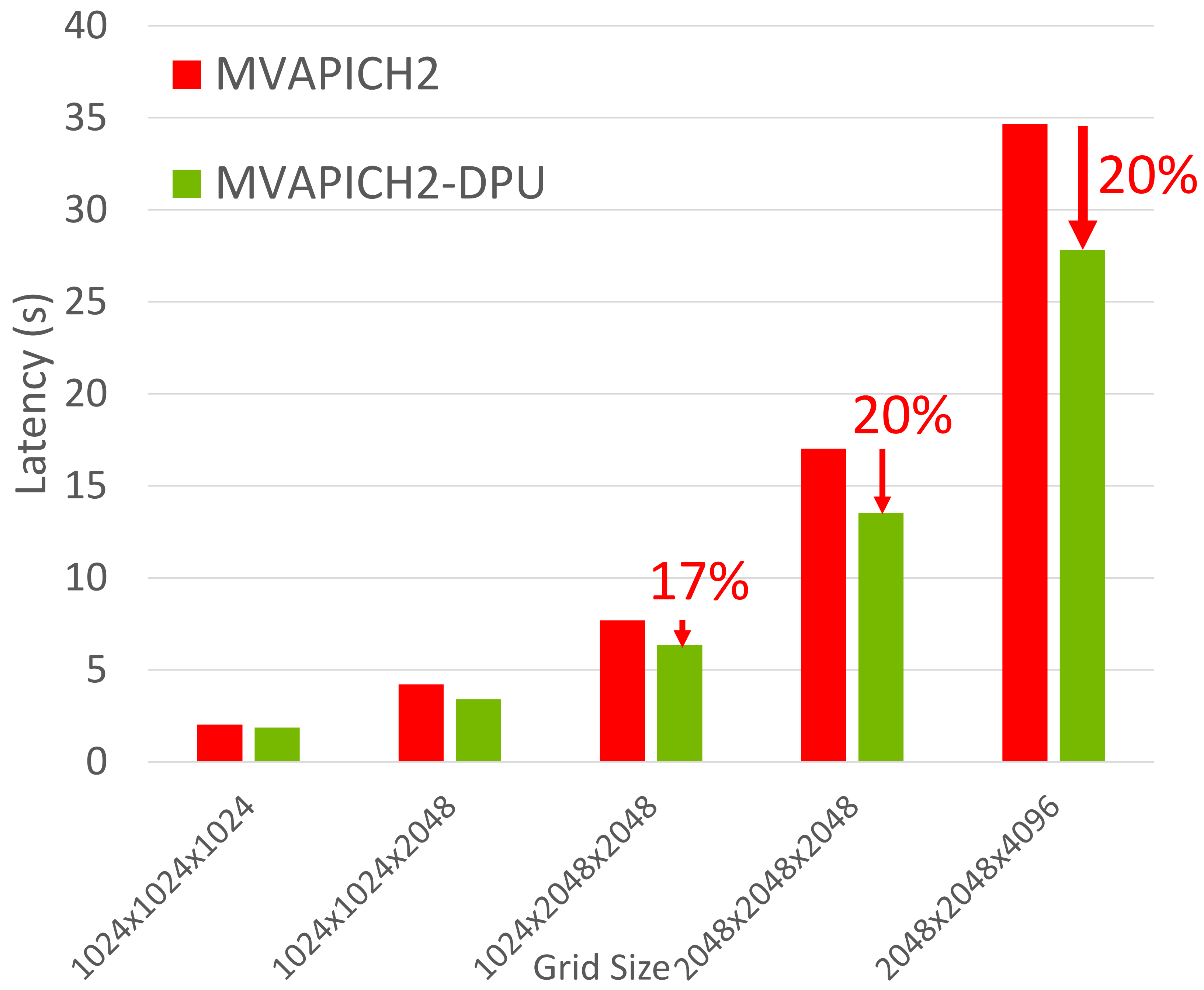
32 Nodes, 32 PPN

Benefits in Total execution time (Compute + Communication)

P3DFFT Application Execution Time (16 nodes)



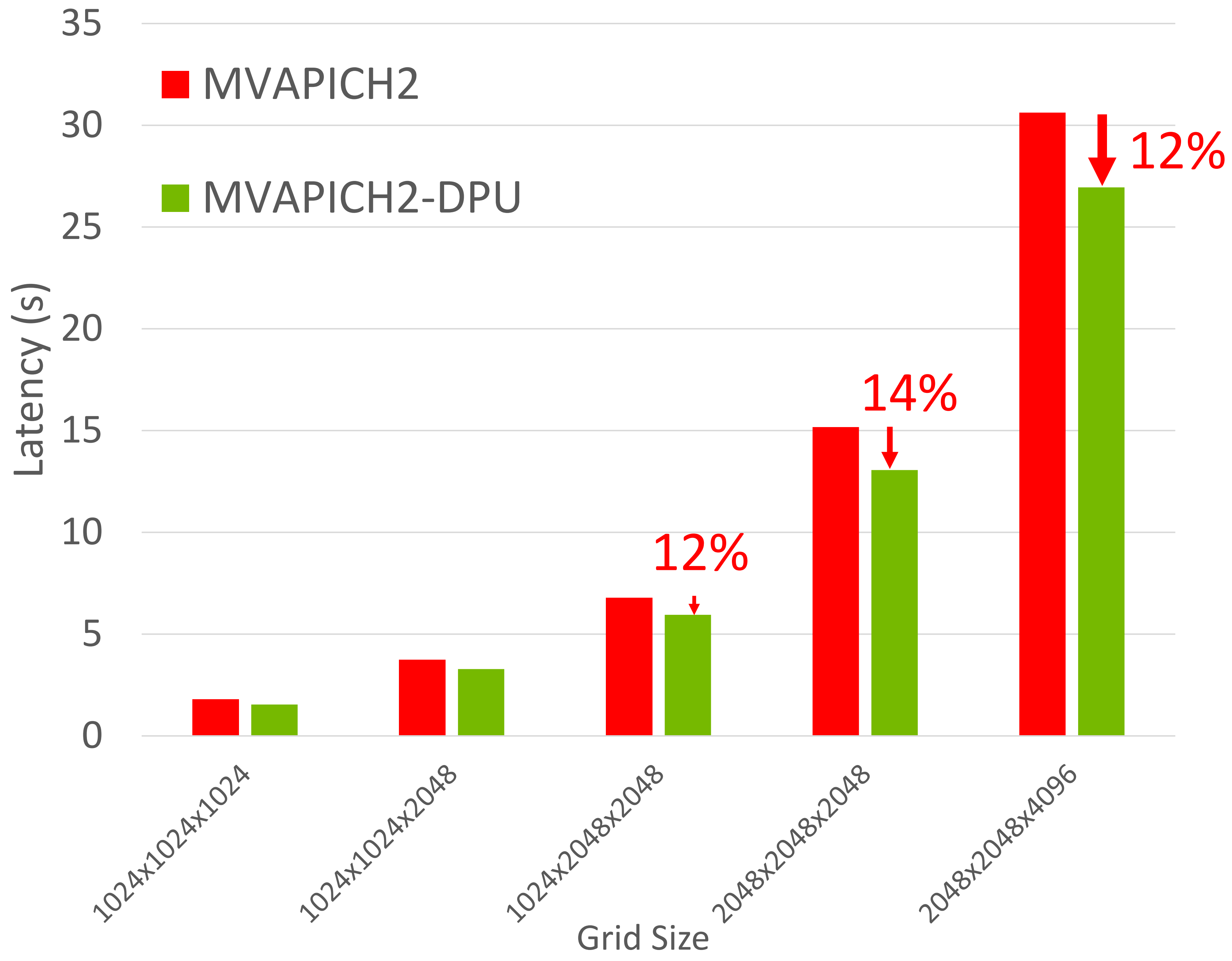
16 Nodes, 16 PPN



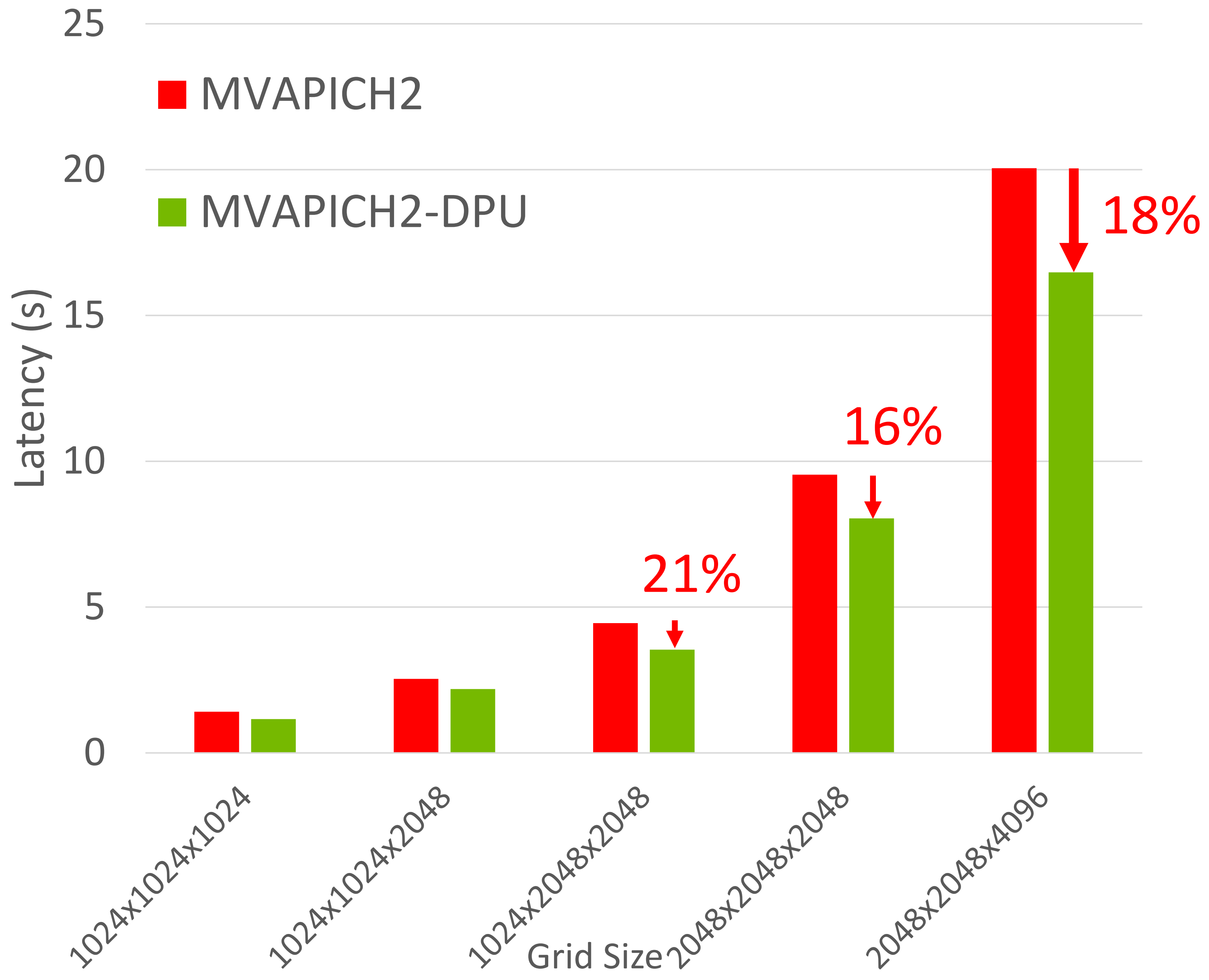
16 Nodes, 32 PPN

Benefits in application-level execution time

P3DFFT Application Execution Time (32 nodes)



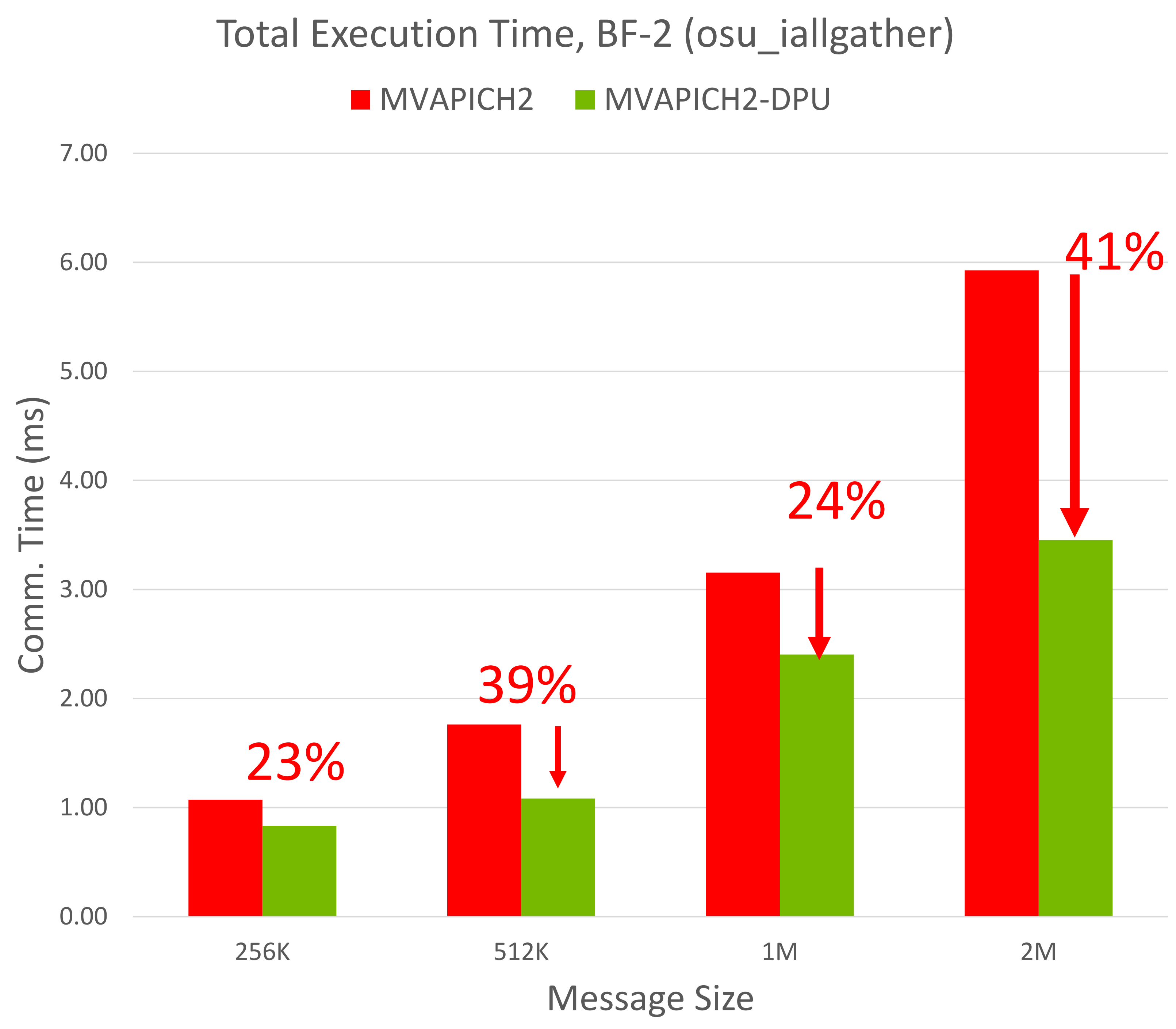
32 Nodes, 16 PPN



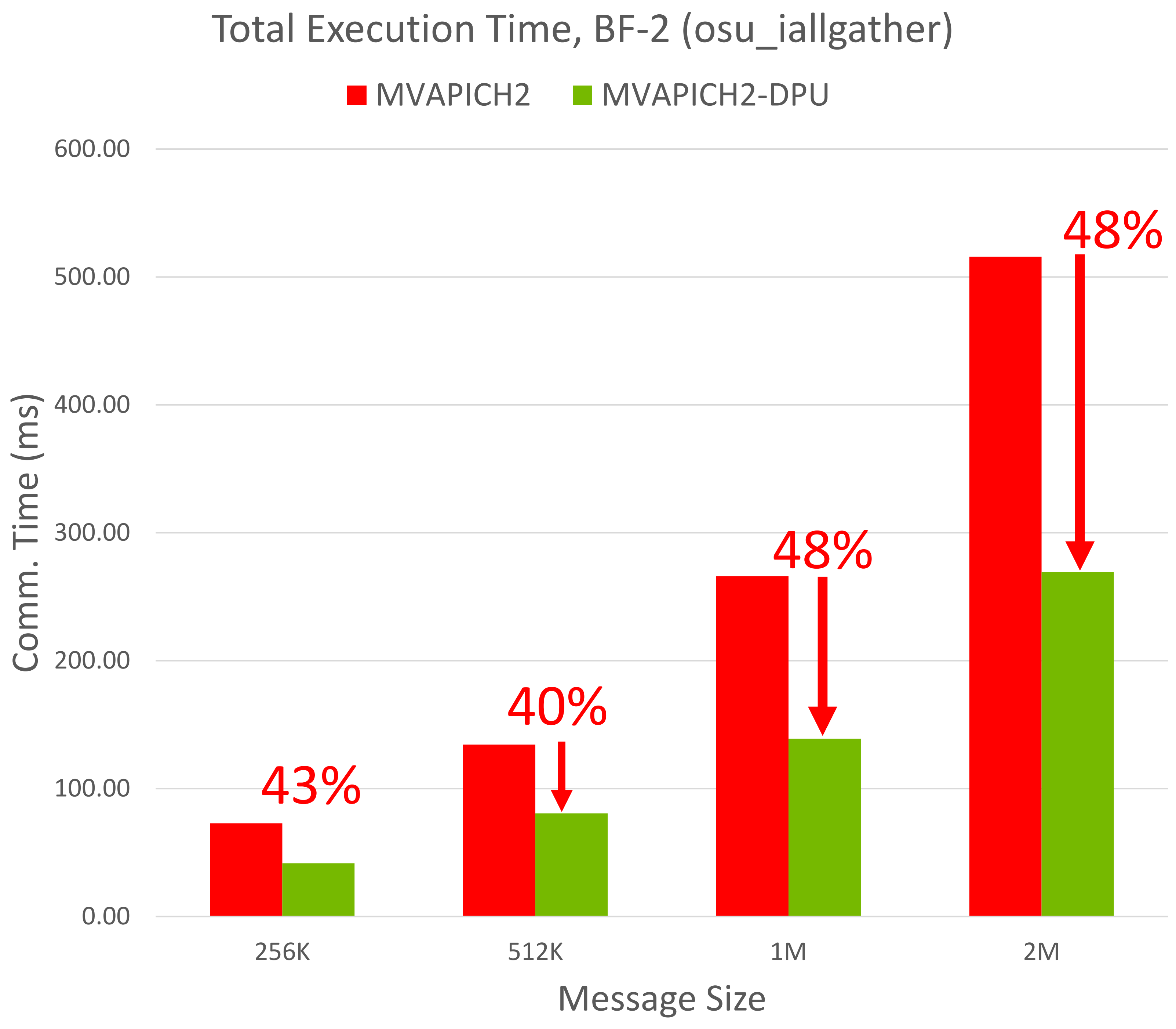
32 Nodes, 32 PPN

Benefits in application-level execution time

Total Execution Time with osu_iallgather (16 nodes)



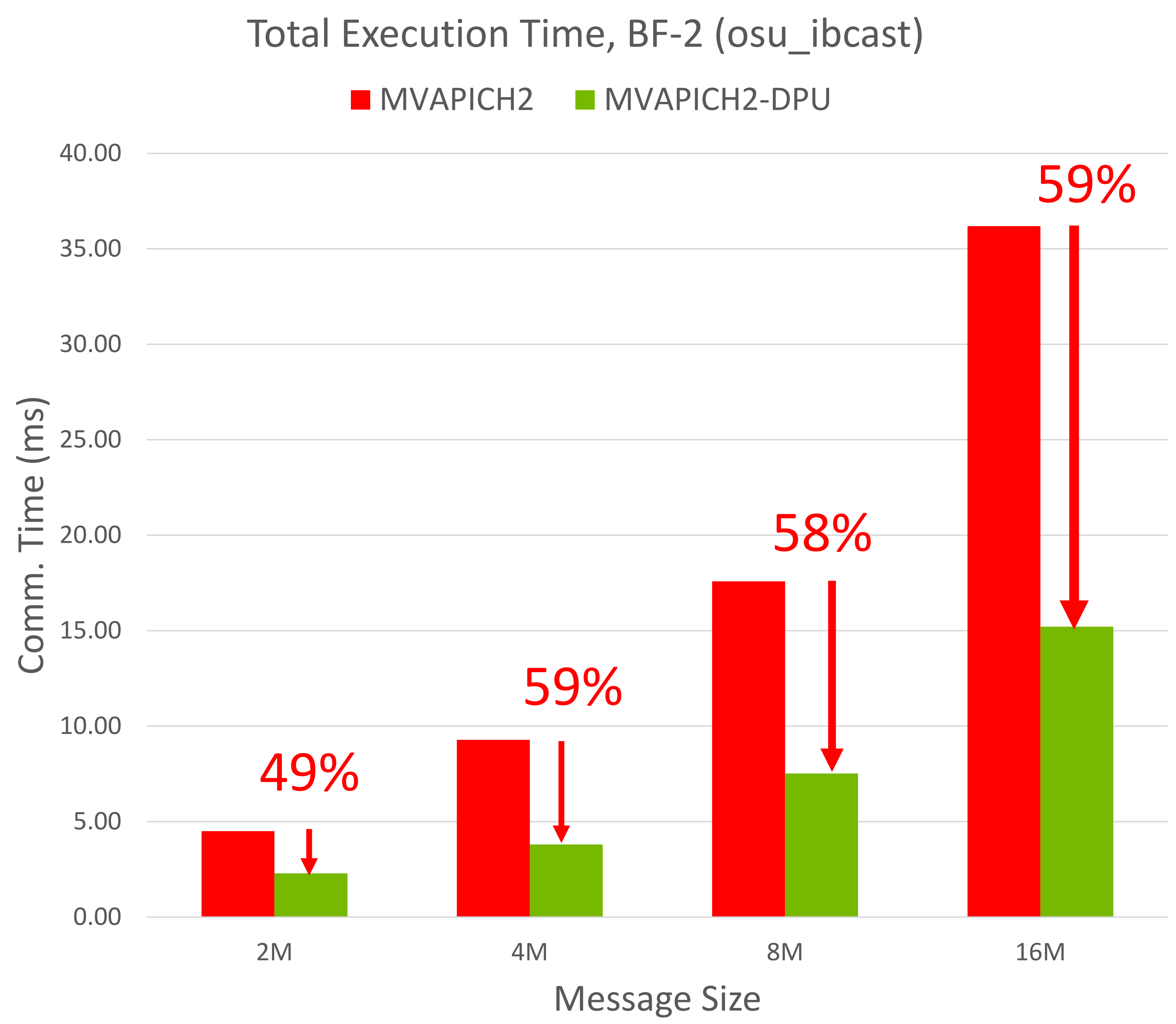
16 Nodes, 1 PPN



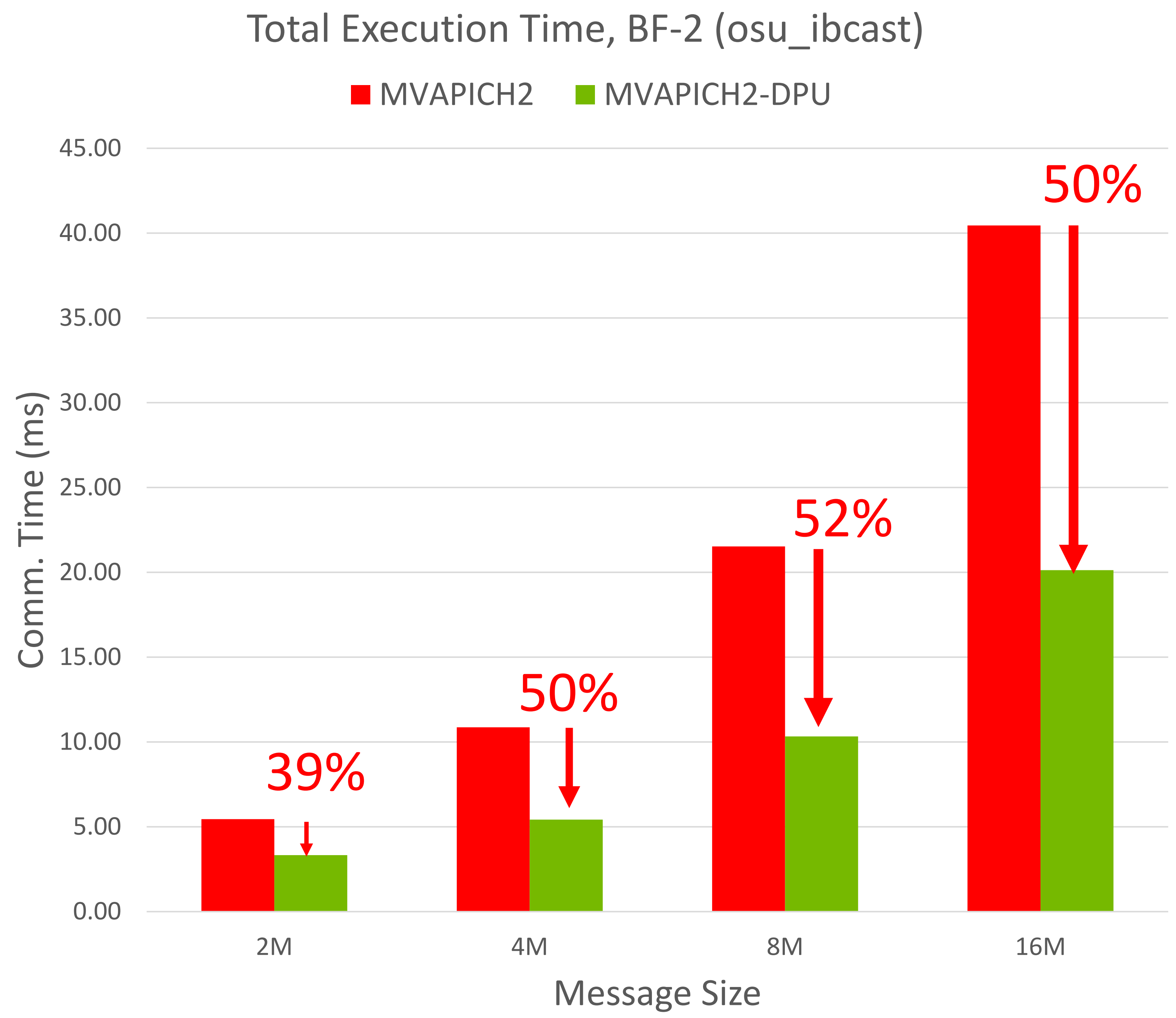
16 Nodes, 16 PPN

Total Execution Time with osu_Iallgather (16 nodes)

Total Execution Time with osu_ibcast (16 nodes)



16 Nodes, 16 PPN

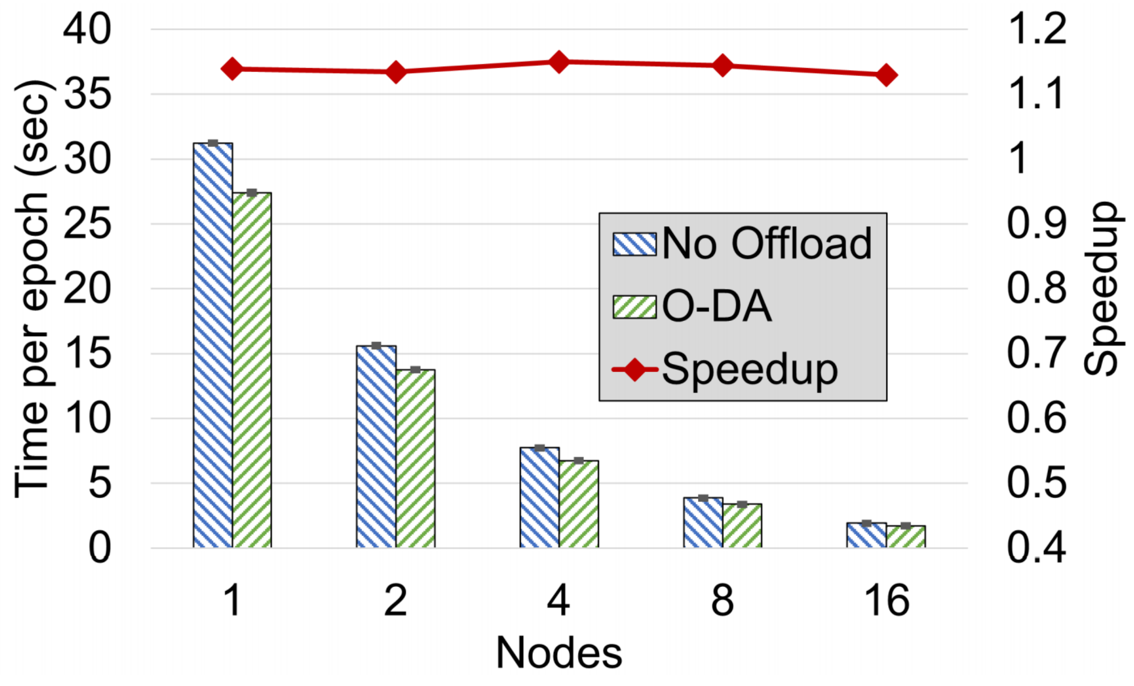


16 Nodes, 32 PPN

Total Execution Time with osu_Ibcast (16 nodes)

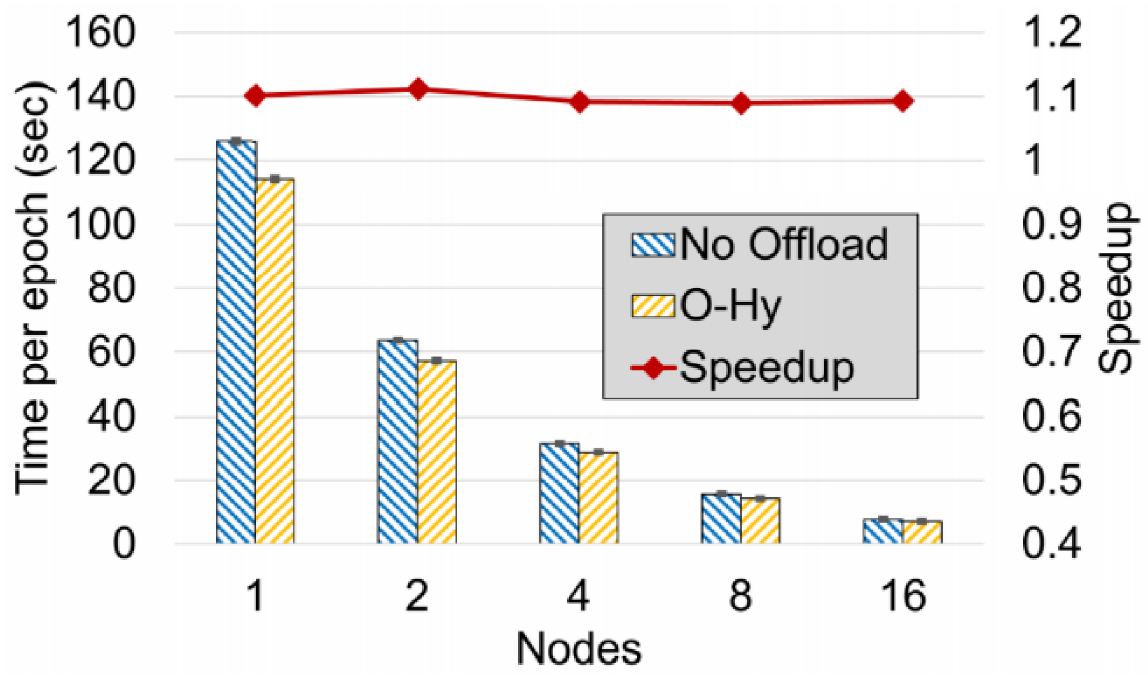
Benefits of SMART NICs to DL Applications

Training ShuffleNet on Tiny ImageNet Dataset



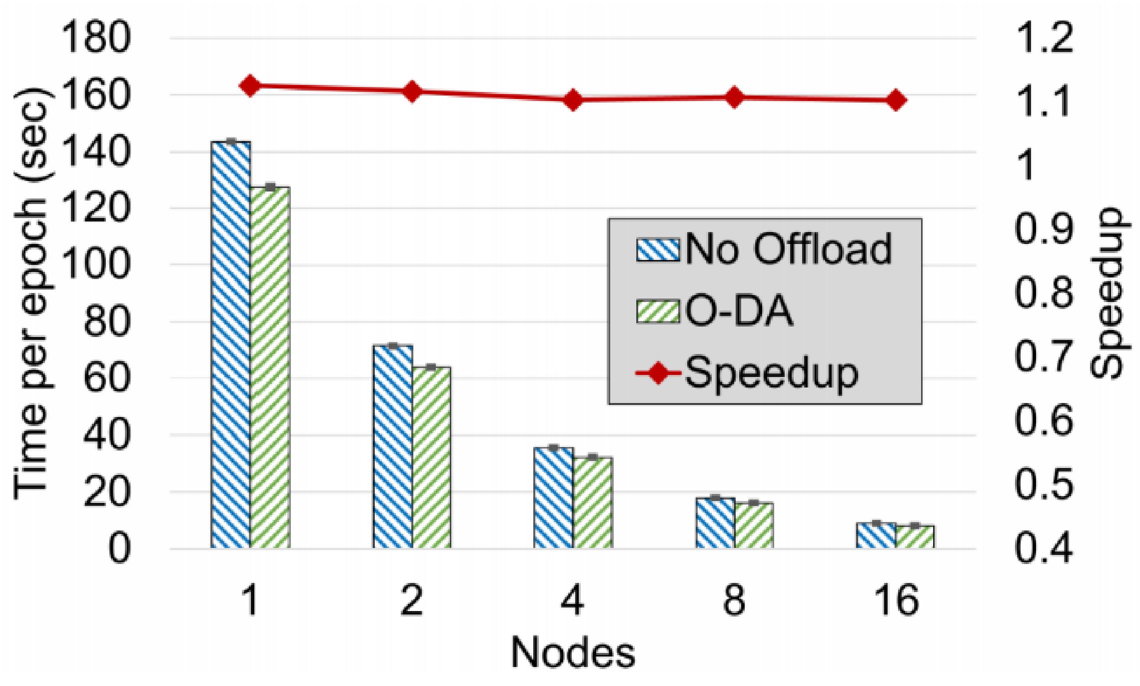
Offload achieves 13.9% speedup on average on 1-16 nodes

Training ResNet-56 on SVHN Dataset



Offload achieves 9.3% speedup on average on 16 nodes

Training ShuffleNet on Tiny ImageNet Dataset



Offload achieves 10.2% speedup on average on 16 nodes

- Everything or Based on the capabilities?
- Offloading compute (as things stand now) – bad idea!
- What is best suited to the capability of the DPU – orchestration of communincation and I/O
  - Offload Data Augmentation (O-DA)
  - Offload Model Validation (O-MV)

A. Jain, N. Alnaasan, A. Shafi, H. Subramoni, D. Panda, “Accelerating CPU-based Distributed DNN Training on Modern HPC Clusters using BlueField-2 DPUs”, HotI28

Packet Processing Engines or General-Purpose Accelerator

- SMART NICs can be used as both PPEs or GPAs
  - Examples of PPEs
    - Hardware Tag Matching to perform rendezvous offload
    - Streaming reduction
  - Examples of GPAs
    - Enhanced Data Type Processing
    - Offloading complex collective communication patterns

Requirements for Next-Generation MPI Libraries

- Message Passing Interface (MPI) libraries are used for HPC and AI applications
- Requirements for a high-performance and scalable MPI library:
  - Low latency communication
  - High bandwidth communication
  - Minimum contention for host CPU resources to progress non-blocking collectives
  - High overlap of computation with communication
- CPU based non-blocking communication progress can lead to sub-par performance as the main application has less CPU resources for useful application-level computation

Can MPI Functions be Offloaded?

- The area of network offloading of MPI primitives is still nascent and cannot be used as a universal solution
- State-of-the-art BlueField DPUs bring more compute power into the network
- Can we exploit additional compute capabilities of modern BlueField DPUs into existing MPI middleware to extract
  - Peak pure communication performance
  - Overlap of communication and computation

For dense non-blocking collective communications?

Programming Models and Tools

- We have not used any specialized tools to utilize SMART NICs
- We see a clear need for a standardized interface
  - OpenSNAPI
- Currently SMART NICs appear as separate hosts to user level libraries
- Can next-gen SMART NICs be enhanced to provide direct access to host memory
  - Allow to initiate transfers on behalf of the host from host memory