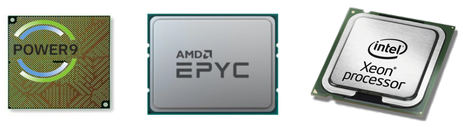
Designing GPU-Aware Collective Communication for Heterogeneous Clusters with Diverse GPUs and Interconnects

Chen-Chun Chen (Advisor: Dhabaleswar K. Panda)
The Ohio State University

Introduction – Trends in Modern HPC Clusters

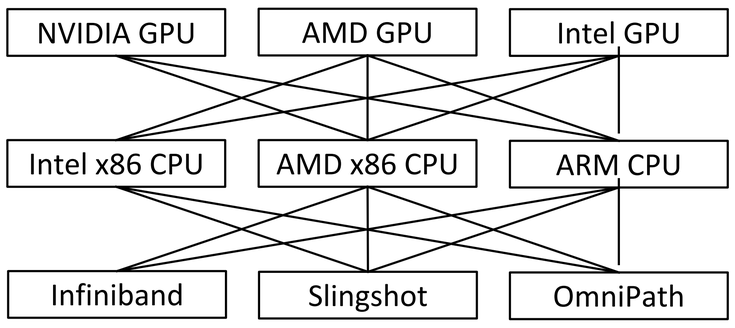


Multi / Many-core Processors:
- X86
- ARM

High Performance Interconnects:
- InfiniBand
- Omni-Path
- Slingshot

<1usec latency,
400Gbps+ Bandwidth

Accelerators (GPUs, FPGA):
High compute power
High peak memory bandwidth
High bandwidth Interconnect:
- NVIDIA NVLink/NVSwitch
- AMD Infinity Fabric
- Intel X^e Link



Introduction – Trends in Modern Large-scale Dense-GPU Systems

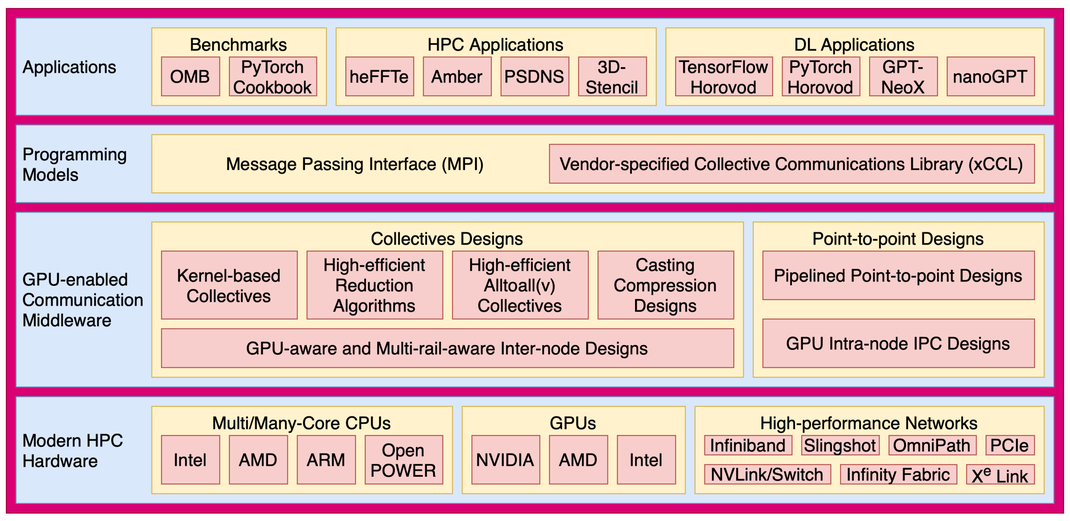
- In addition to computation power, applications also rely on efficient communication libraries in modern GPU systems
- HPC Applications
  - heFFTe: Highly Efficient FFTs for Exascale
    - Utilizes Alltoall communication for medium to large message sizes (MB-GB)
  - Amber: a suite of biomolecular simulation programs, such as proteins, nucleic acids, and lipids
    - Utilizes Allreduce communication for medium message sizes (MB)
- AI Applications/Frameworks
  - Horovod with TensorFlow, PyTorch
  - PyTorch
    - Distributed Data Parallelism (DDP) / Fully Sharded Data Parallel (FSDP) framework
    - LLM training and inference, MoE, GPT-like models
    - Utilizes both reduction-based and data-movement-based communications for medium to large message sizes (MB-GB)

Challenge and Problem Statement

How do we design a **GPU-Aware Collective** Communication middleware to fully support the **heterogeneous** clusters with diverse **GPUs** and **Interconnects**?

1. How can the zero-copy load-store capabilities of GPU IPC be leveraged to create high-performance, IPC-enhanced Alltoall collective operations?
2. What novel GPU-aware Allreduce algorithm can be developed to minimize latency and maximize scalability on large GPU clusters?
3. Can unified Allreduce algorithms be developed to efficiently operate across heterogeneous CPU and GPU architectures while adapting to hierarchical, multi-rail interconnect environments?

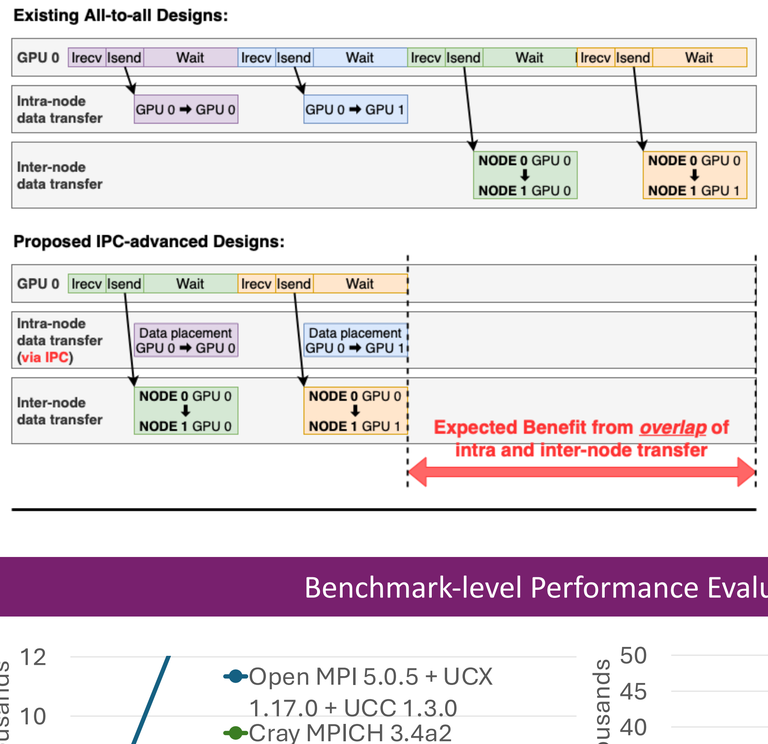
Research Framework



1. IPC-based Alltoall Designs for Dense GPU Systems

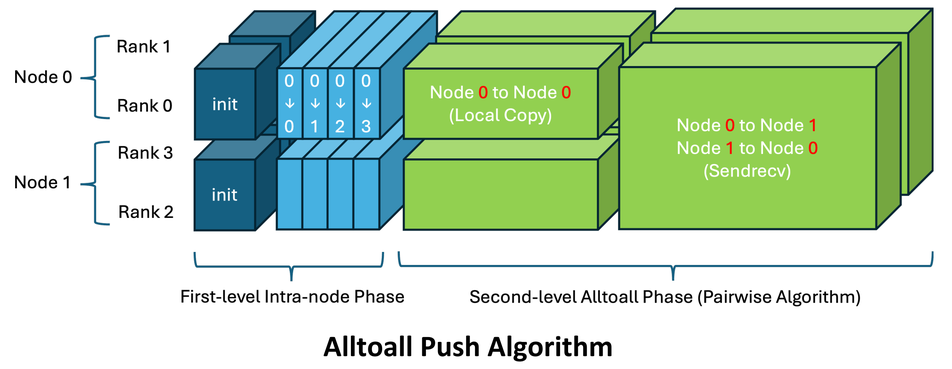
Proposed IPC-advanced Algorithm:

- The proposed IPC-advanced designs provided overlap potential of intra-node and inter-node communication through utilizing zero-copy load-store IPC mechanisms.

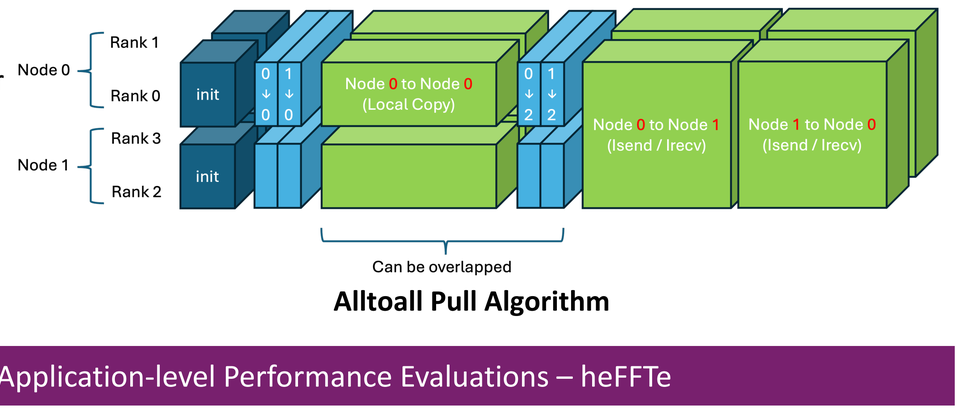


Multi-leader Two-level Alltoall Algorithm:

- **Push Algorithm:**
  - Steps:
    - **Each rank pushes data** directly to the GPU buffers of peers (via IPC)
    - After all intra-node pushes finish**, leader ranks** perform inter-node Alltoall
  - Limitation: No overlap possible due to **sync barrier** between phases
- **Pull Algorithm:**
  - Steps:
    - Inter-node **Irecv** launched early for each peer
    - Intra-node Pull: use IPC to **copy required segments** into a **persistent GPU buffer**
    - Once ready, launch **Isend**
  - Enables fine-grained **overlap** of intra- and inter-node stages

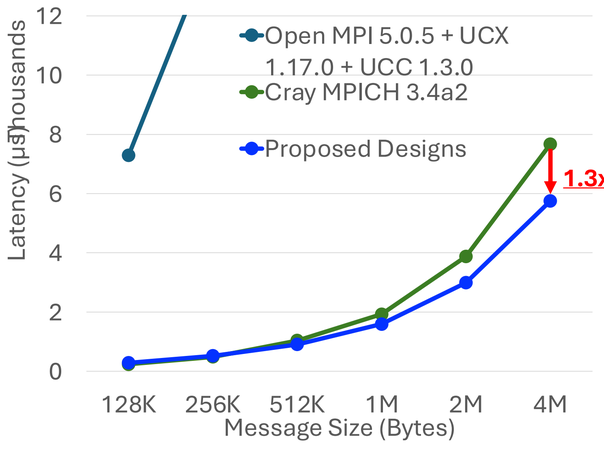


Alltoall Push Algorithm

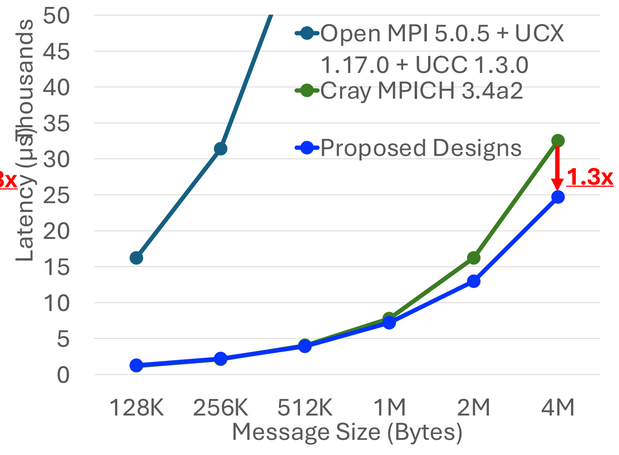


Alltoall Pull Algorithm

Benchmark-level Performance Evaluations

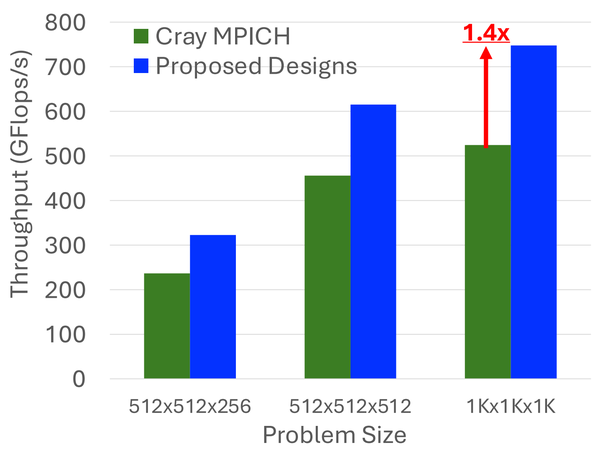


Isambard-AI – 8 nodes (32 GPUs)

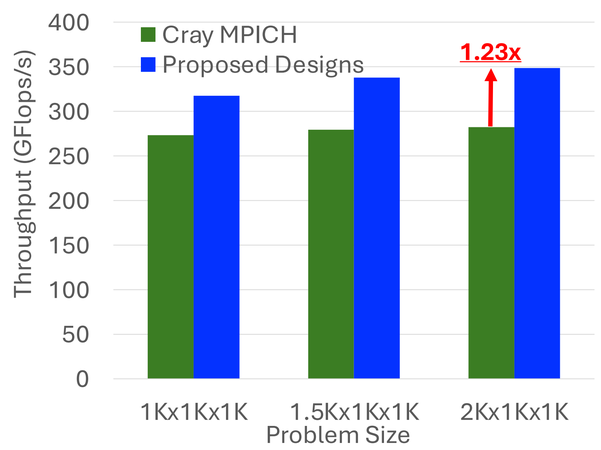


Isambard-AI – 8 nodes (32 GPUs)

Application-level Performance Evaluations – heFFTe



Isambard-AI – 8 nodes (32 GPUs)



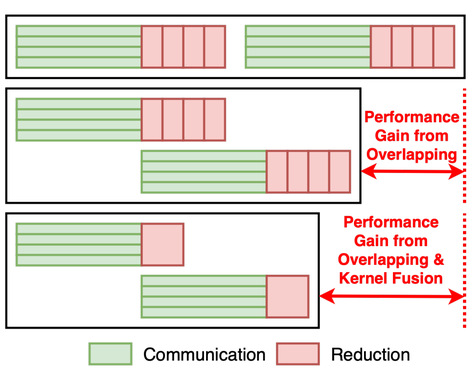
Frontier – 8 nodes (64 GPUs)

Publications

1. **Chen-Chun Chen**, Jinghan Yao, Hari Subramoni, and Dhabaleswar K. Panda, "Design and Optimization of GPU-Aware MPI Allreduce Using Direct Sendrecv Communication, " Accepted for ICPP 2025.
2. **Chen-Chun Chen**, Jinghan Yao, Lang Xu, Hari Subramoni, and Dhabaleswar K. Panda, "Unified Designs of Multi-rail-aware MPI Allreduce and Alltoall Operations Across Diverse GPU and Interconnect Systems, " *IPDPS 2025*.
3. **Chen-Chun Chen**, Goutham Kalikrishna Reddy Kuncham, Hari Subramoni, and Dhabaleswar K. Panda, "Design and Implementation of Kernel-based MPI Reduction Operations for Intel GPUs, " *HiPC 2024*.
4. **Chen-Chun Chen**, Goutham Kalikrishna Reddy Kuncham, Pouya Kousha, Hari Subramoni, and Dhabaleswar K. Panda, "Design and Implementation of an IPC-based Collective MPI Library for Intel GPUs, " *PEARC 2024*.
5. **Chen-Chun Chen**, Kawthar Shafie Khorassani, Pouya Kousha, Qinghua Zhou, Jinghan Yao, Hari Subramoni, and Dhabaleswar K. Panda, "MPI-xCCL: A Portable MPI Library over Collective Communication Libraries for Various Accelerators, " *IPDRM 2023, held in conjunction with SC*.
6. **Chen-Chun Chen**, Kawthar Shafie Khorassani, Goutham Kalikrishna Reddy Kuncham, Rahul Vaidya, Mustafa Abduljabbar, Aamir Shafi, Hari Subramoni, and Dhabaleswar K. Panda, "Implementing and Optimizing a GPU-aware MPI Library for Intel GPUs: Early Experiences, " *CCGRID 2023*.
7. **Chen-Chun Chen**, Kawthar Shafie Khorassani, Quentin Anthony, Aamir Shafi, Hari Subramoni, and Dhabaleswar K. Panda, "Highly Efficient Alltoall and Alltoallv Communication Algorithms for GPU Systems, " *HCW 2022, held in conjunction with IPDPS*.

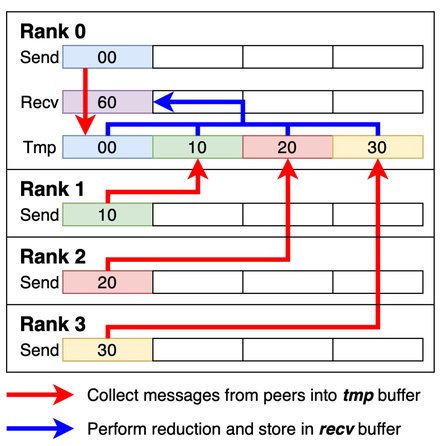
2. Efficient Kernel-based Direct Allreduce Designs

- **Direct Sendrecv algorithm:**
  - Inspired by the Alltoall communication pattern
  - Aim to saturate communication bandwidth and improve scalability by concurrently collecting messages from peer processes
  - Prevent data dependencies like the Ring or Recursive Doubling Algorithm
- Where to store the collected messages for subsequent computation?
  - Reuse the persistent *tmp* buffer
- Optimization:
  - Overlapping: overlap communication and computation phases
  - Kernel fusion: minimize kernel launch and memory access overhead

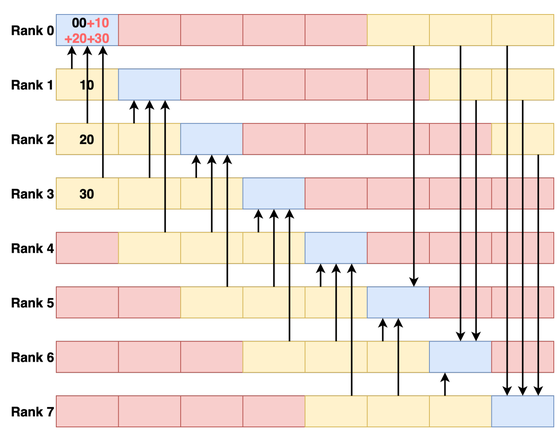


Timeline illustration of Optimization

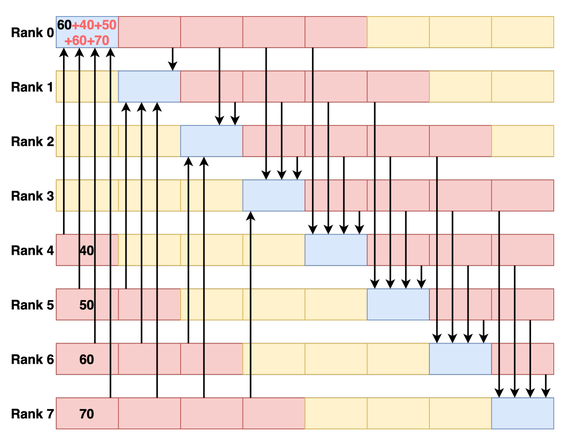
- **Throttling mechanism:**
  - Group send and receive calls into smaller batches to reduce contention
- Example: 8 processes using a throttle factor of 4, resulting in 2 batches, see Rank 0
  - Step 1: collect data from ranks 1 through 3 and performs a local reduction that includes its own data
  - Step 2: gather the remaining messages from ranks 4 through 7 and aggregates them with the previously reduced result from step 1



Data Placement

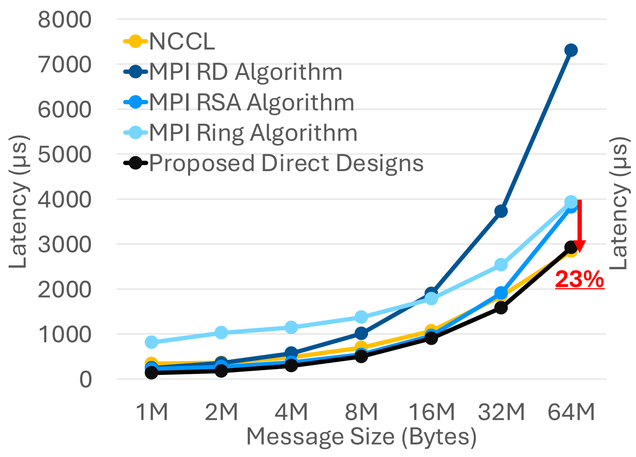


Direct Algorithm with Throttling - Step 1

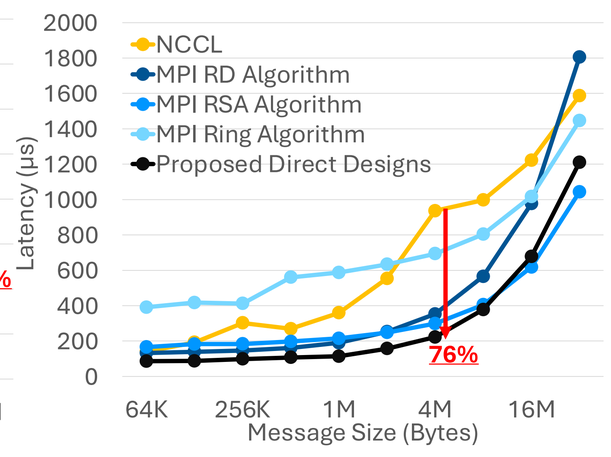


Direct Algorithm with Throttling - Step 2

Benchmark-level Performance Evaluations

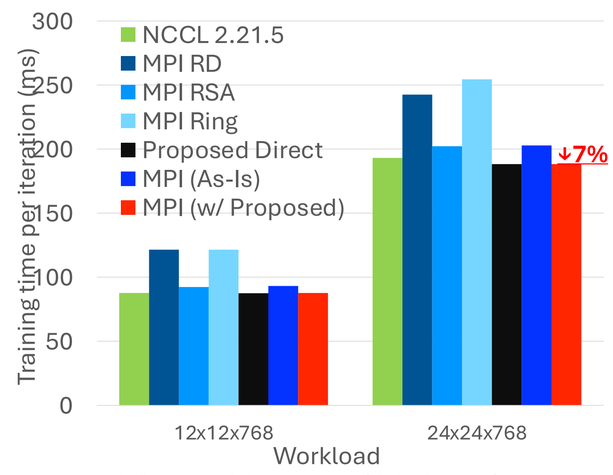


Vista – 32 nodes (32 GPUs)



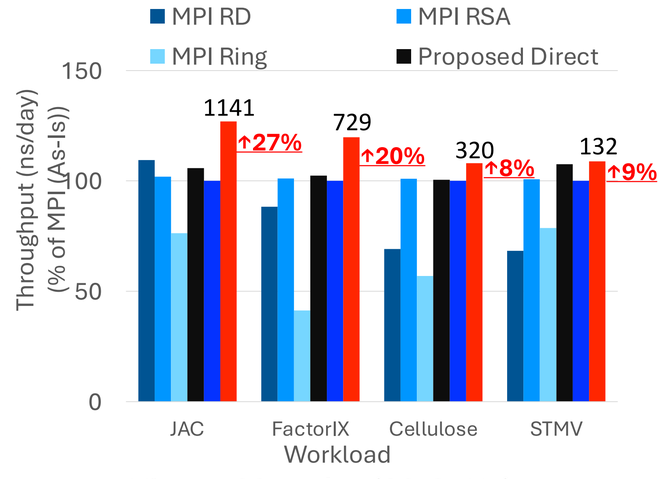
Delta-AI – 16 nodes (64 GPUs)

Application-level Evaluations – nanoGPT



Vista – 32 nodes (32 GPUs)

Application-level Evaluations – Amber

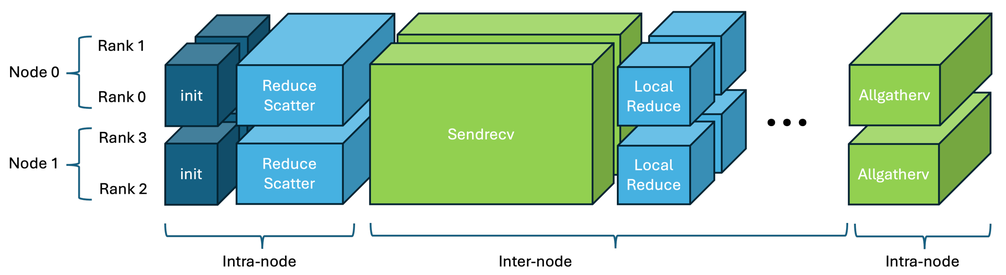


Vista – 32 nodes (32 GPUs)

3. Multi-rail-aware Allreduce Designs across Diverse GPU and Interconnect Systems

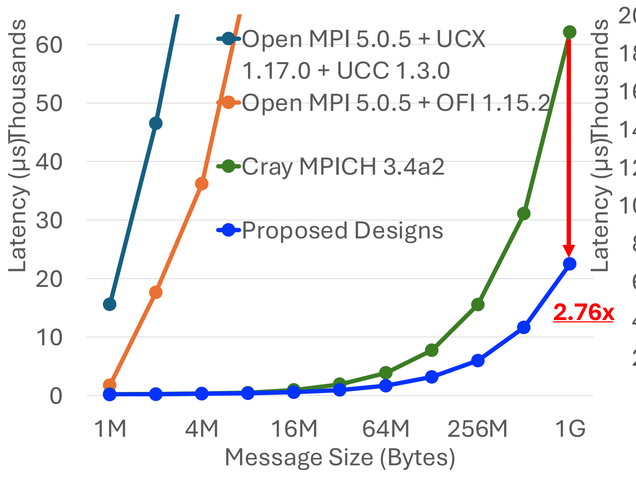
- **Persistent GPU** buffer: avoid CPU staging
- **Two-level hierarchy** (intra + inter node): leader-based communication to isolate fast local ops from slower inter-node transfers
- **Multi-leader** strategy: **overlap** intra-node (ReduceScatter) with inter-node (Allreduce), and then intra-node (Allgather)
- Unified for Allreduce
  - Diverse GPU vendors: **NVIDIA**, **AMD**, Intel
  - Diverse networks: **Infiniband**, **Slingshot**, **Omni-Path**
  - Multiple algorithms: RD, RSA

- **Two-Level Allreduce Structure:**
  - **Intra-node** phase: use GPU kernels for ReduceScatter and Allgather, employ IPC to access peer GPU buffers
  - **Inter-node** phase: ranks with same local rank form leader groups, each group runs a second-level Allreduce
  - **Persistent** GPU buffer
  - Early-triggered and **pipelined** optimization
    - Divide data into num_chunks

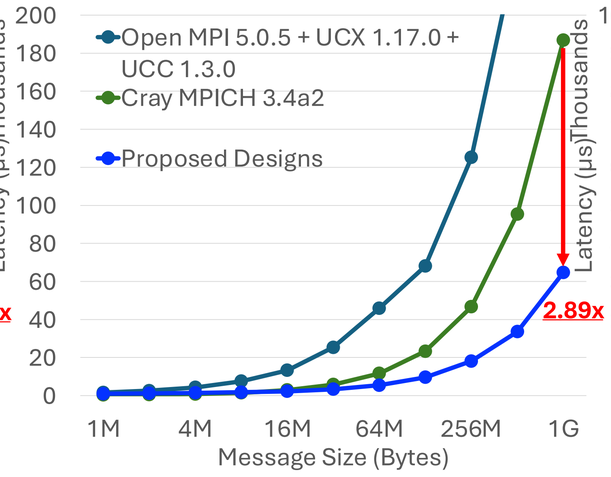


Dataflow of the proposed multi-Leader two-Level Allreduce algorithm

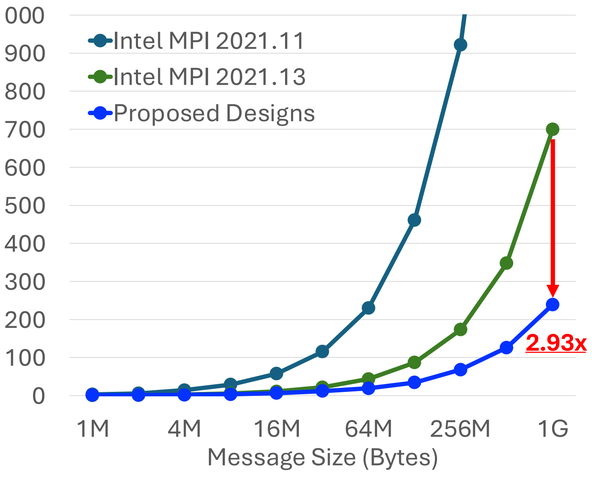
Benchmark-level Performance Evaluations



Isambard-AI – 8 nodes (32 GPUs)

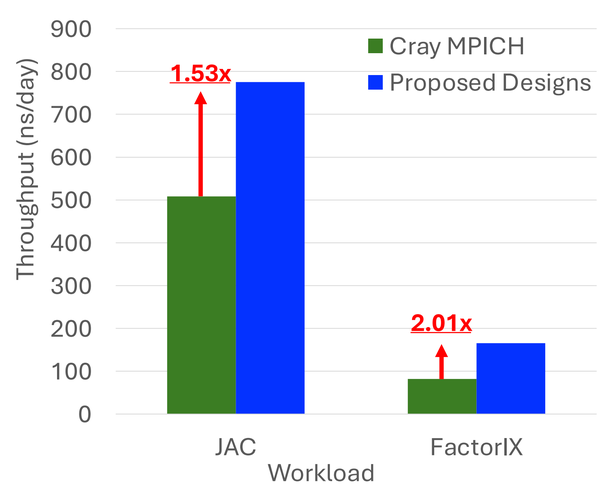


Frontier – 8 nodes (64 GPUs)



Stampede3 – 8 nodes (64 GPUs)

Application-level Evaluations – Amber



Cardinal – 8 nodes (32 GPUs)

Impact on HPC Community

- MVAPICH2-GDR 2.3.5 (12/11/2020)
- MVAPICH2-GDR 2.3.6 (08/12/2021)
  - Architecture detection and tuning
- MVAPICH2-GDR 2.3.7 (05/27/2022)
  - IPC-based Alltoall Designs for Dense GPU Systems
- MVAPICH-Plus 4.0 a, b, rc, GA (07/26/2024 - 12/20/2024)
  - GPU-aware MPI Designs for Novel Intel GPUs
- MVAPICH-Plus 4.1 rc, GA (06/09/2025, 08/01/2025)
  - Multi-rail-aware Allreduce Designs across Diverse GPU and Interconnect Systems
  - Efficient Kernel-based Direct Allreduce Designs
- Significantly contributed to the design, development, and testing of many generations of MVAPICH-GDR (2.3.{5-7}) and MVAPICH-Plus series

Ongoing Work and Future Research Directions

- Redcuce_scatter and Allgather Designs for Fully Sharded Data Parallel Training
  - FSDP framework decomposes this workflow into MPI_Reduce_scatter and MPI_Allgather, aligning communication with the shard-based memory layout.
  - Optimize with the kernel designs, IPC for intra-node data exchanges, direct RDMA for inter-node transfers, and hierarchical multi-rail pipelines to balance load.
- GPU-aware Accelerated Processing Unit Supports
  - APUs integrate CPU cores and GPU compute units on a single silicon die, offering unified memory spaces and low-latency data sharing.
  - NVIDIA's Grace Hopper and AMD's Ryzen APUs.
  - Extend our GPU-aware collective framework to fully leverage APU features.
    - Enable direct load/store access between CPU and GPU buffers.
    - When data reside in CPU memory, reduction operations can be offloaded to GPU kernels.

THE OHIO STATE UNIVERSITY

MVAPICH