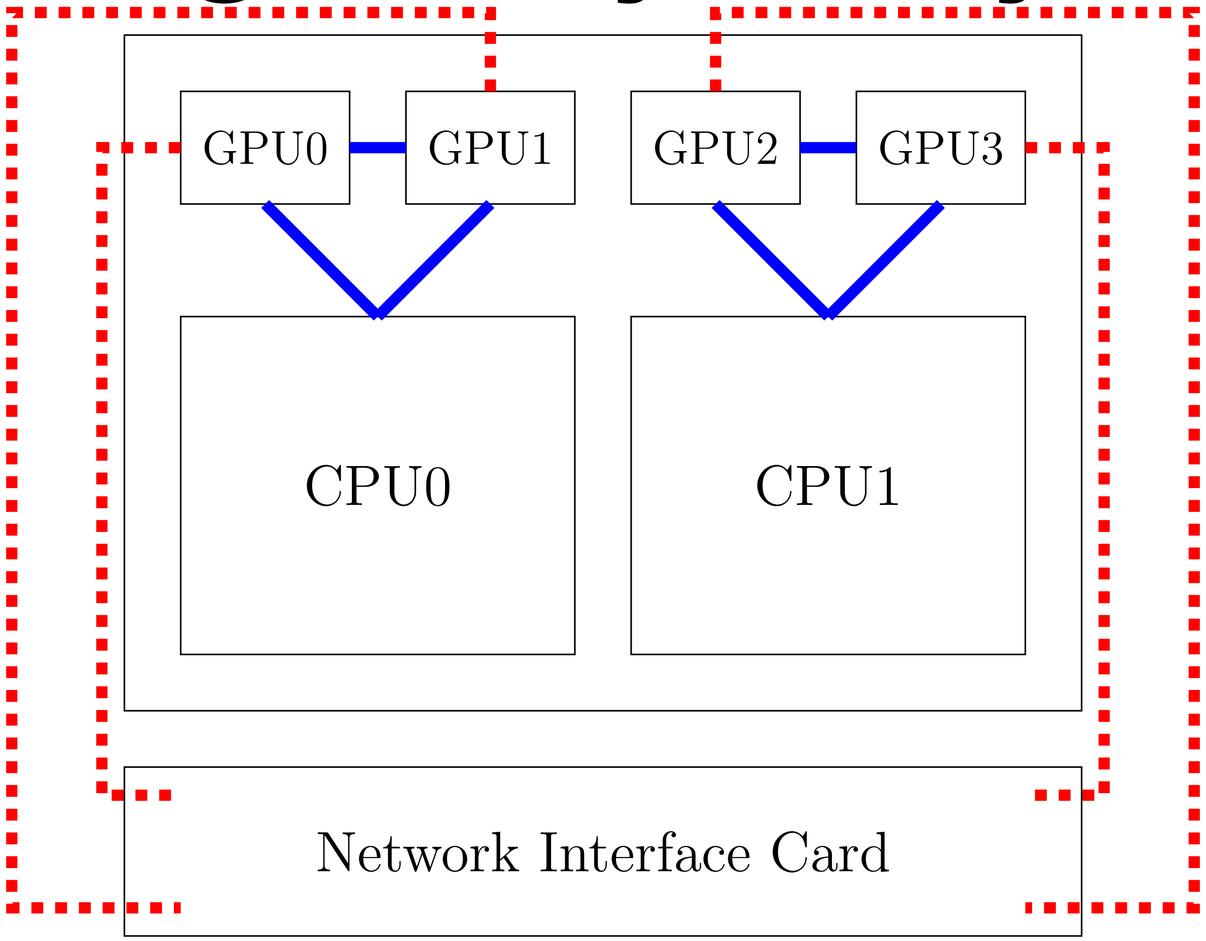
Introduction to Parallel Processing

Lecture 25 : Algorithms for Heterogeneous Systems

11/30/2022

Professor Amanda Bienz

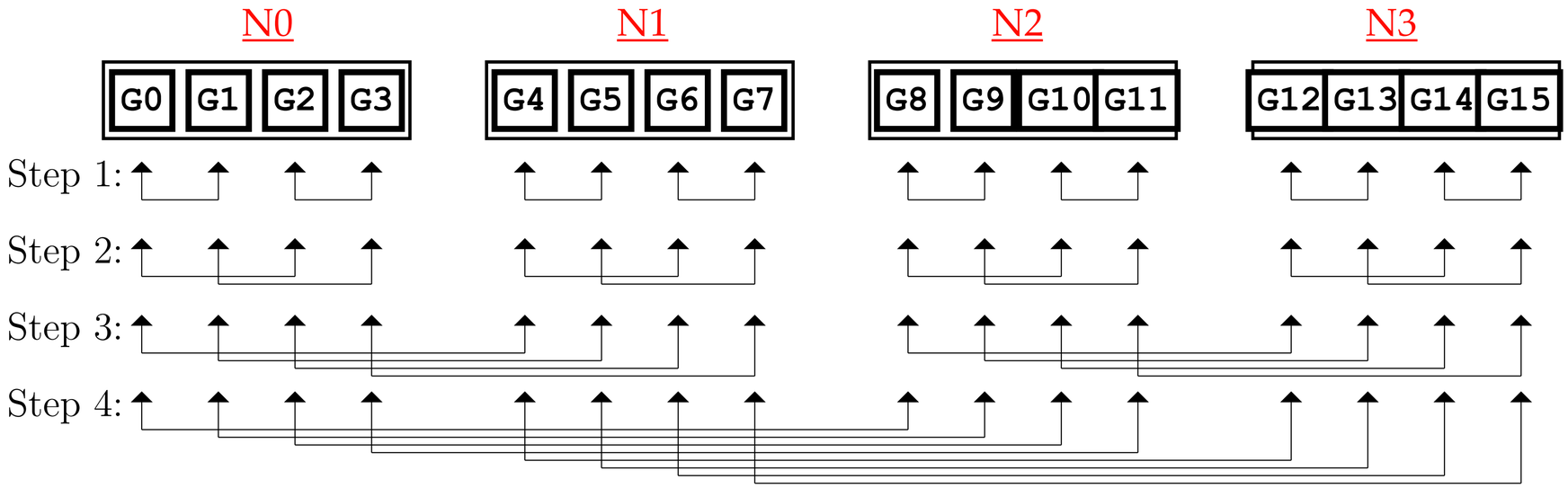
Heterogeneity in Systems



How should we communicate on these systems?

Heterogeneous MPI_Allreduce

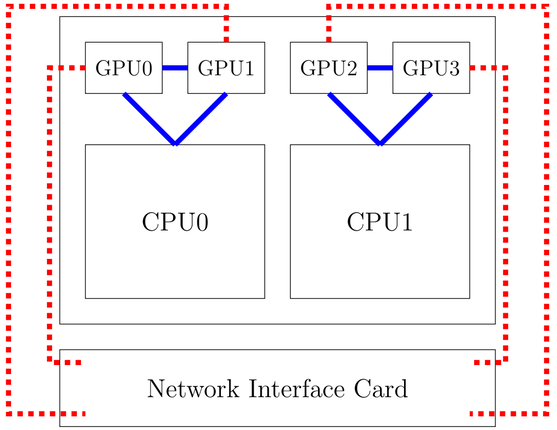
Recursive Doubling



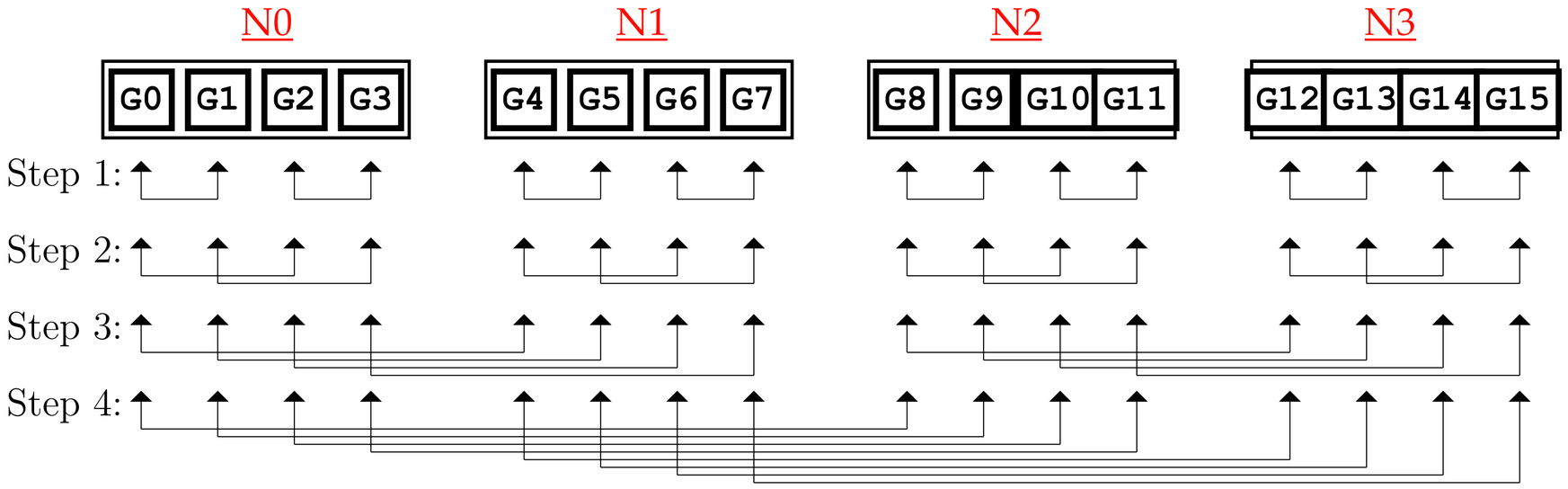
- Standard algorithm for small MPI_Allreduce calls
- Now, need to communicate between GPUs

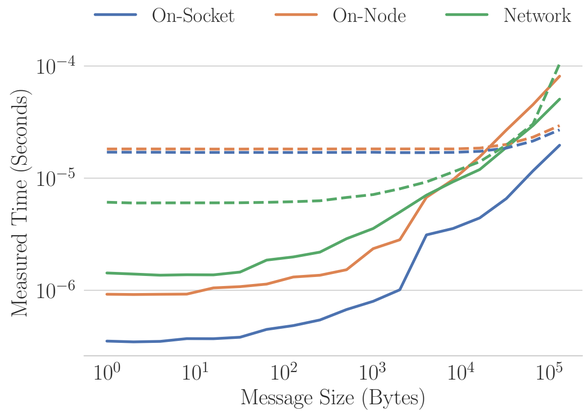
Cuda-Aware Allreduce

- `cudaMalloc((void**)&data, size)`
  `MPI_Allreduce(data, size, …)`



Recursive Doubling

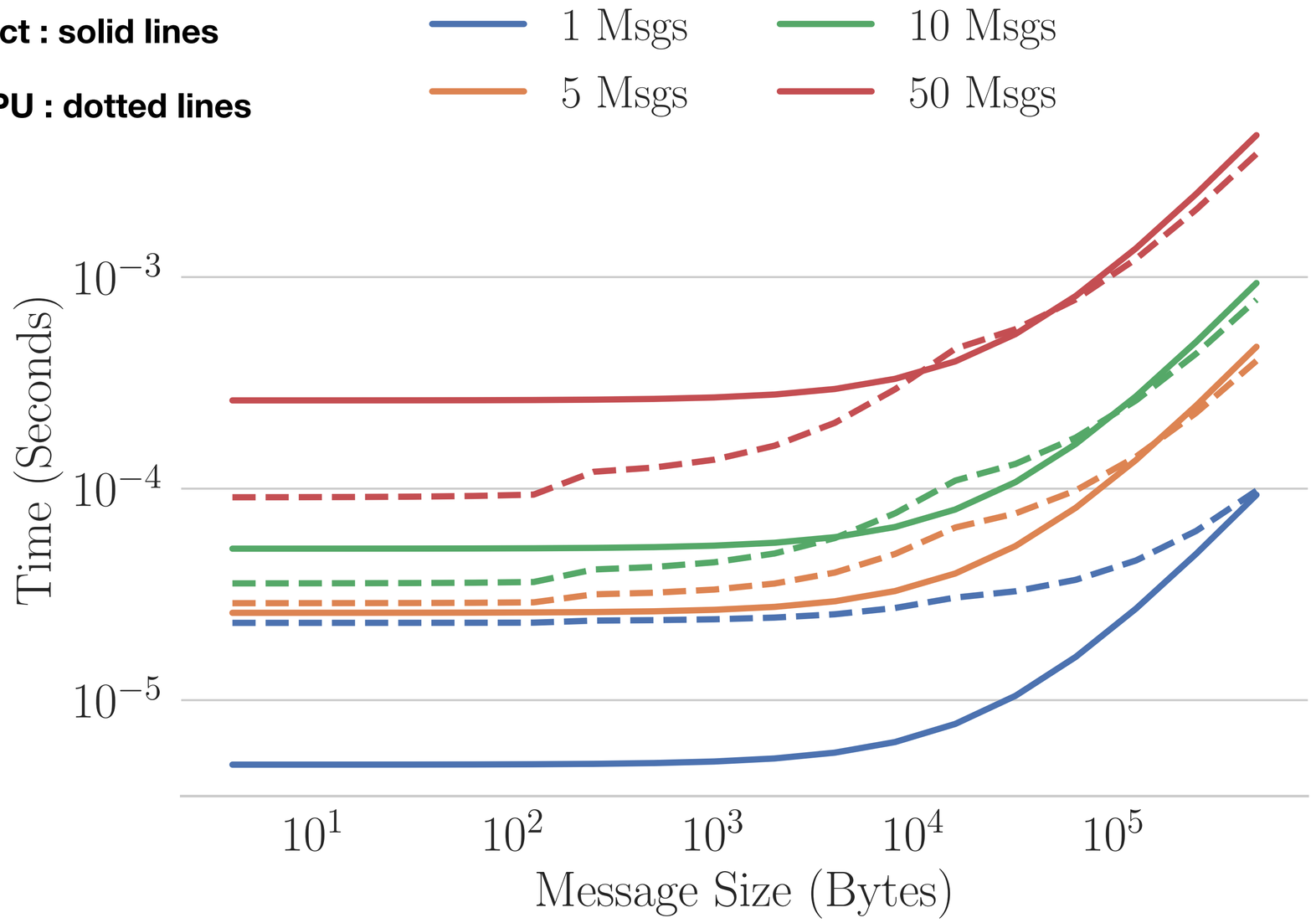




Reminder :
Multiple Messages Cheaper to Copy to CPU

GPUDirect : solid lines

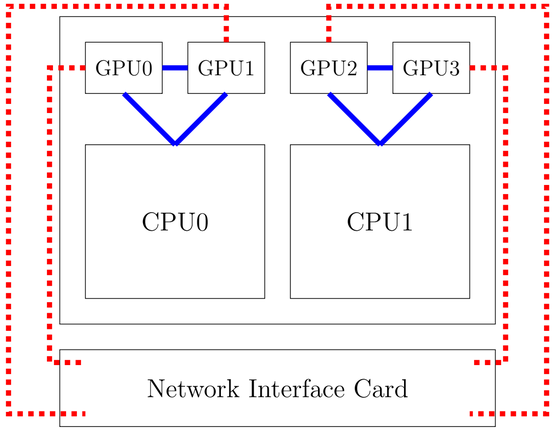
Copy to CPU : dotted lines



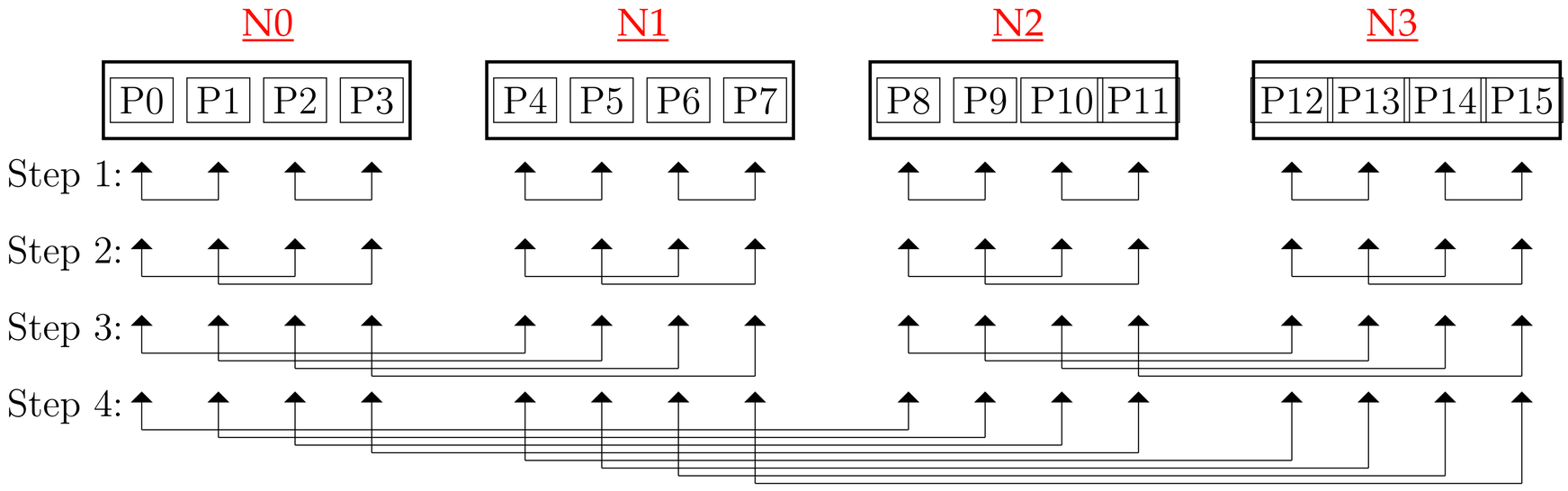
Copy To CPU Allreduce

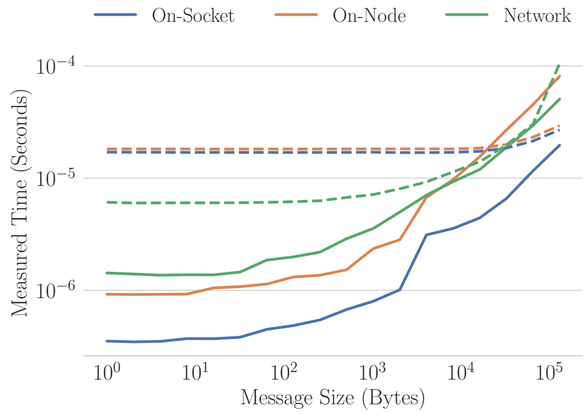
- `cudaMalloc((void**)&d_data, size)`
  `cudaMallocHost((void**)&h_data, size)`

- `cudaMemcpy(h_data, d_data, size, cudaMemcpyDeviceToHost)`
  `MPI_Allreduce(h_data, size, …)`
  `cudaMemcpy(d_data, h_data, size, cudaMemcpyHostToDevice)`

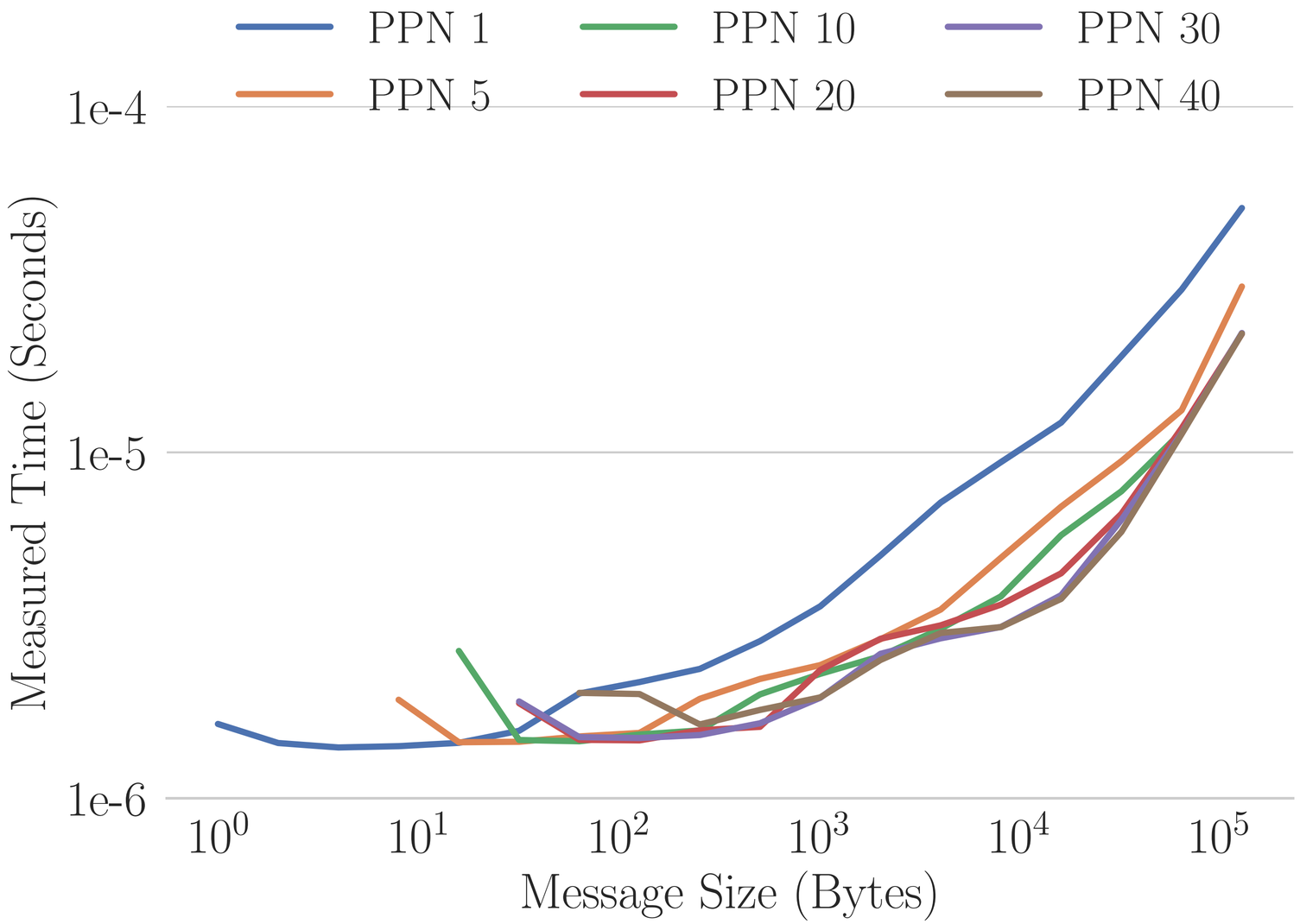


Recursive Doubling



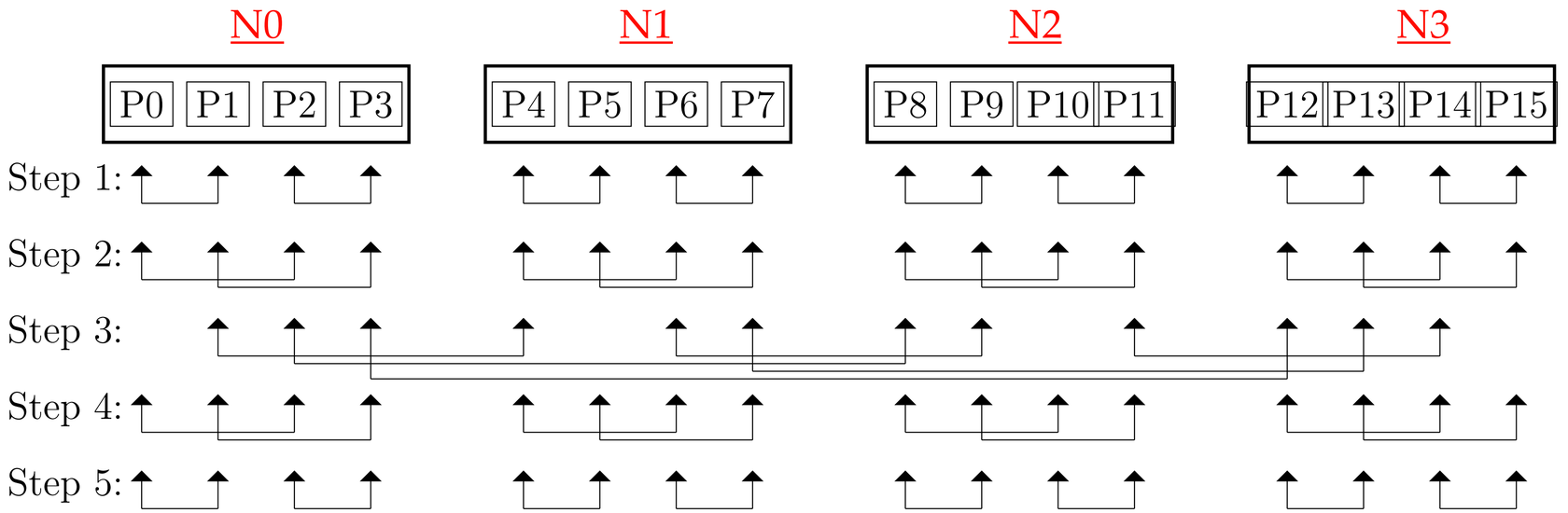


Reminder :
Dozens of available CPU cores per node

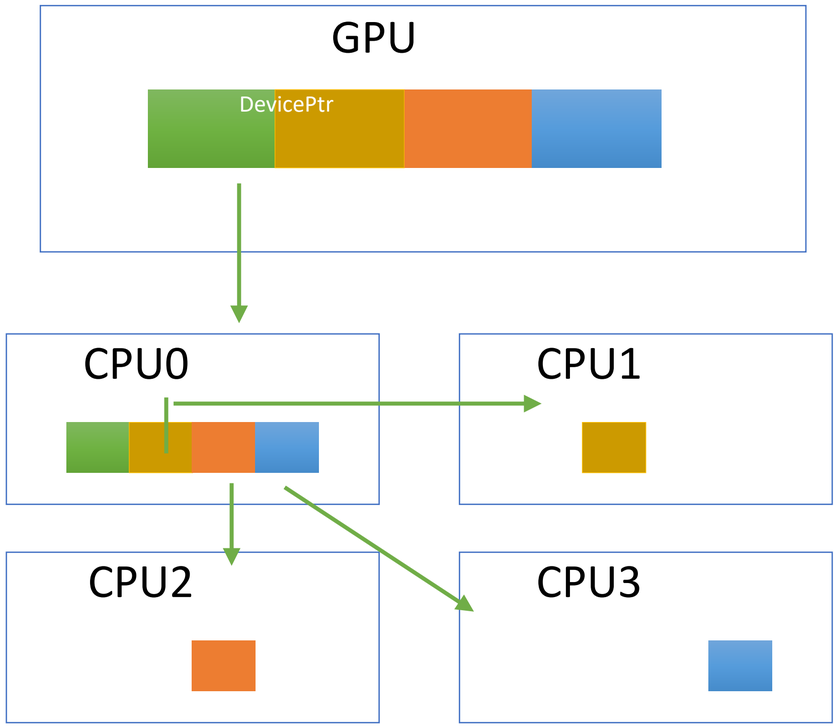


Optimizing Copy-to-CPU Algorithm

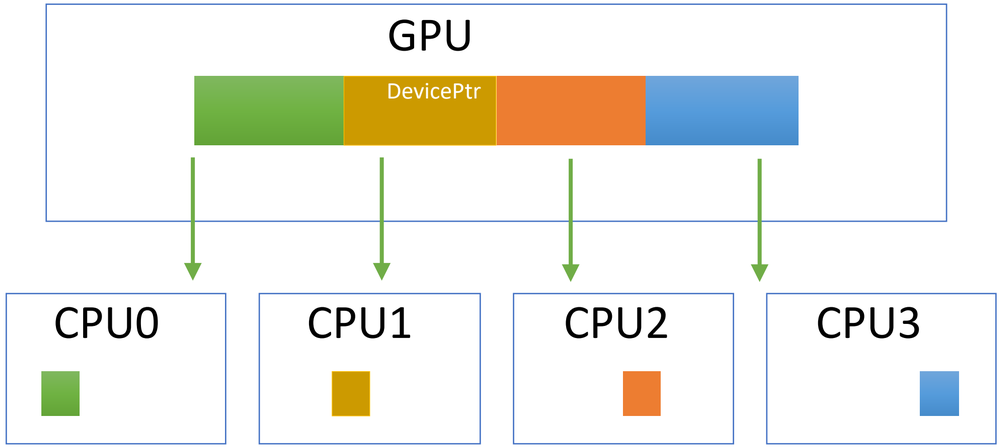
Node-Aware Parallel (NAP) Algorithm



Using Multiple CPU Cores Per GPU

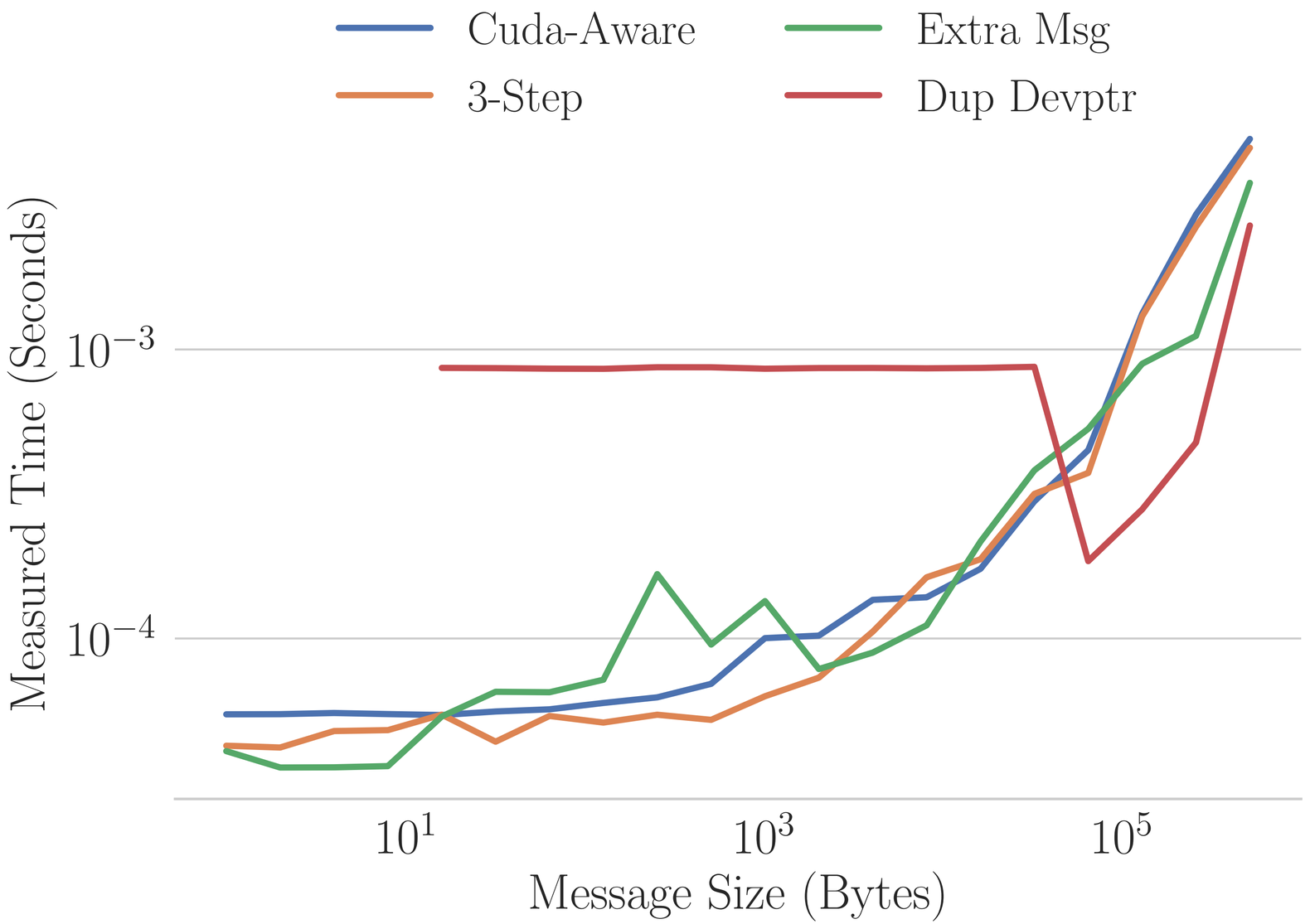


Extra Message Approach

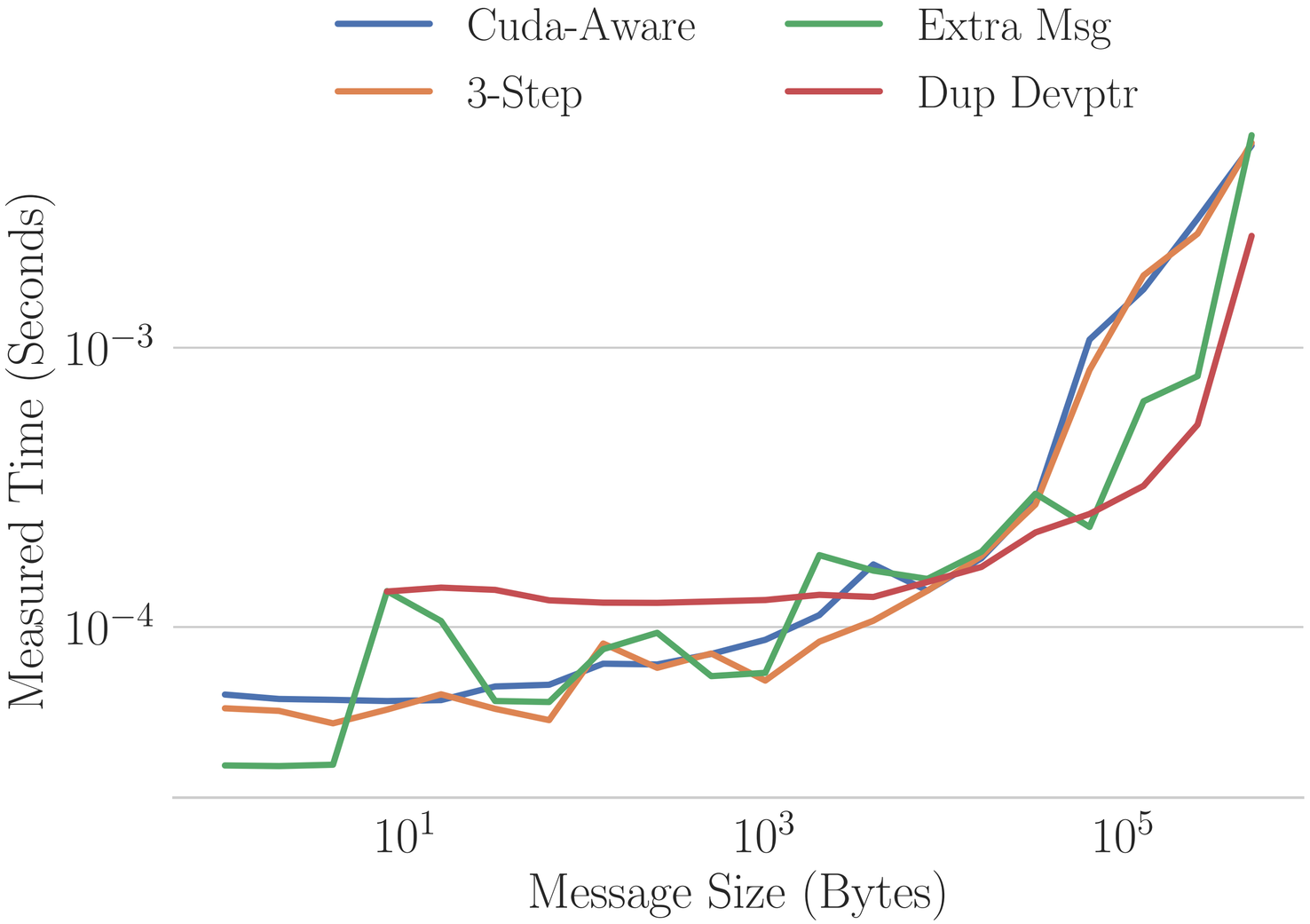


Duplicate Device Pointer Approach

Lassen MPI_Allreduce



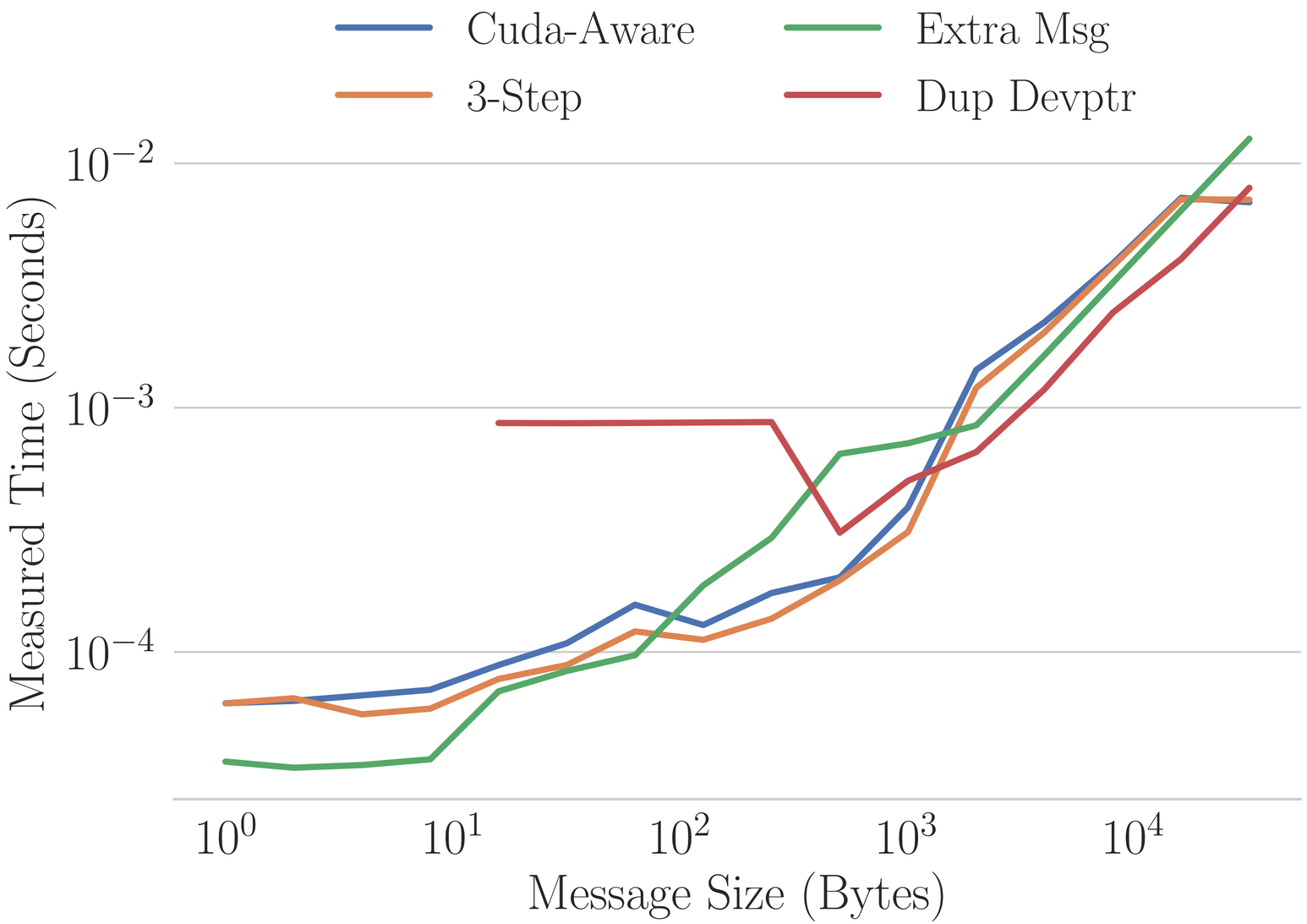
Summit MPI_Allreduce



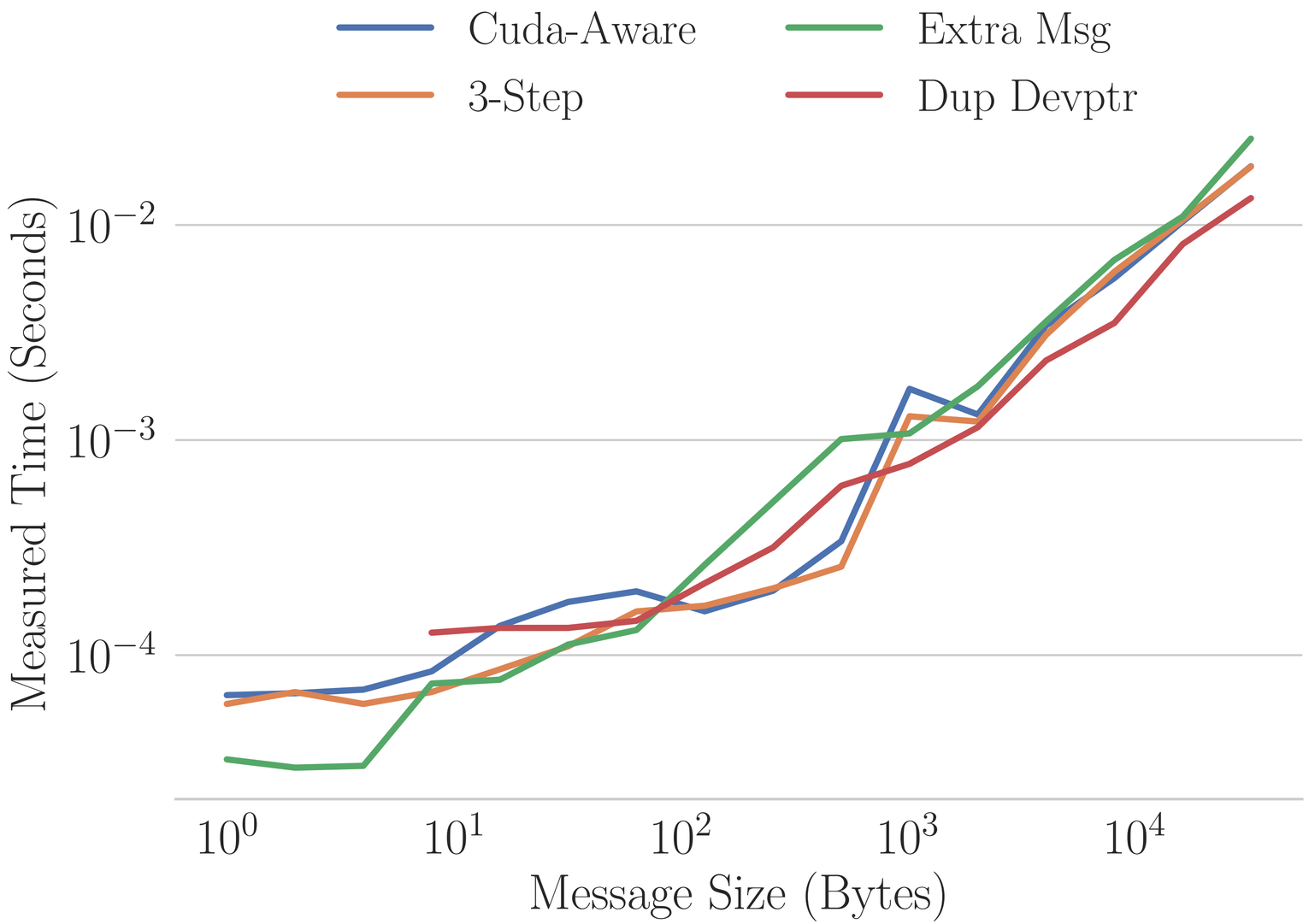
MPI_Allreduce Computation

- Local computation on GPU or CPU?
- What are the benefits of using the GPU?
- What are the benefits of using the CPU?

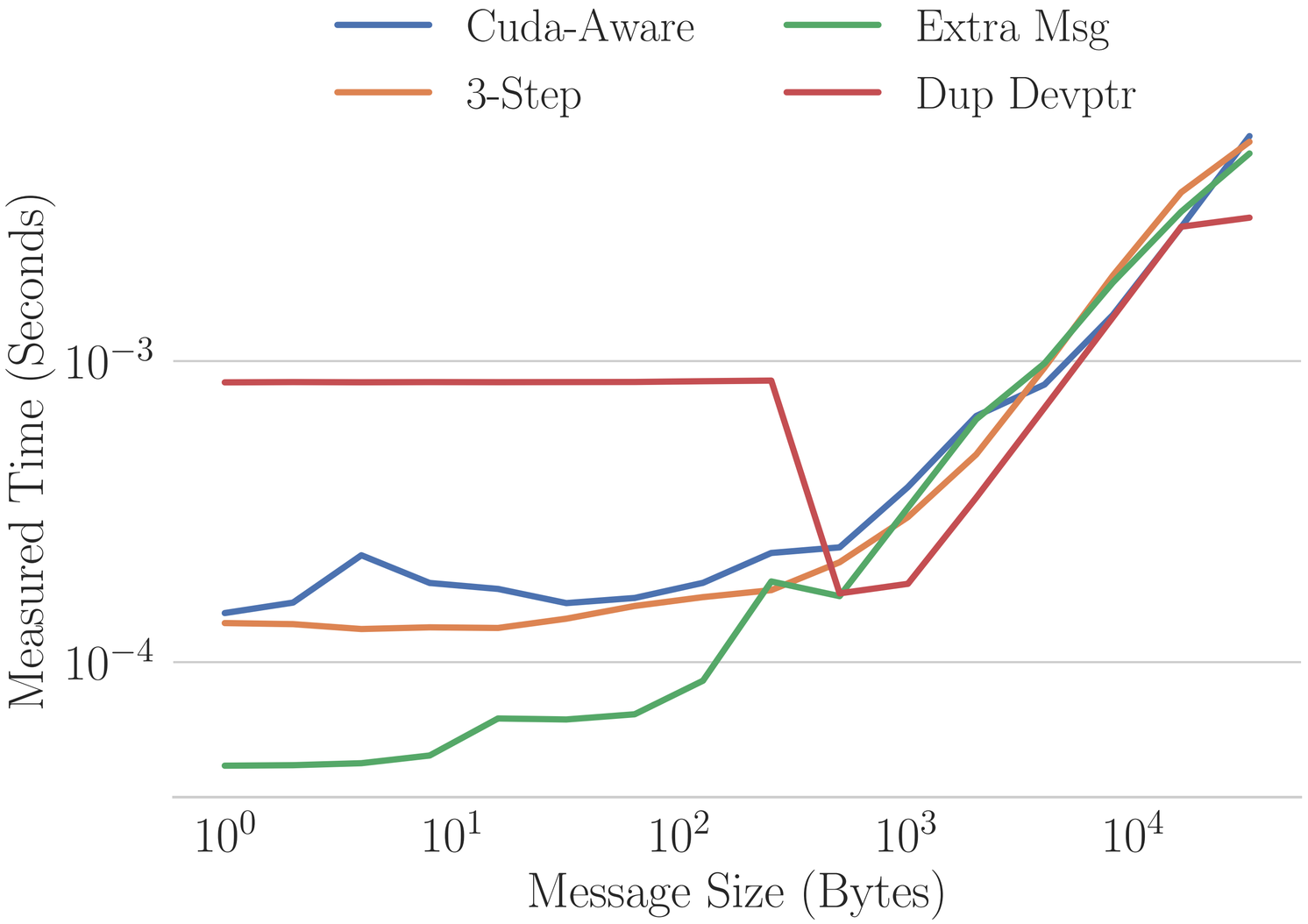
Lassen : MPI_Alltoall



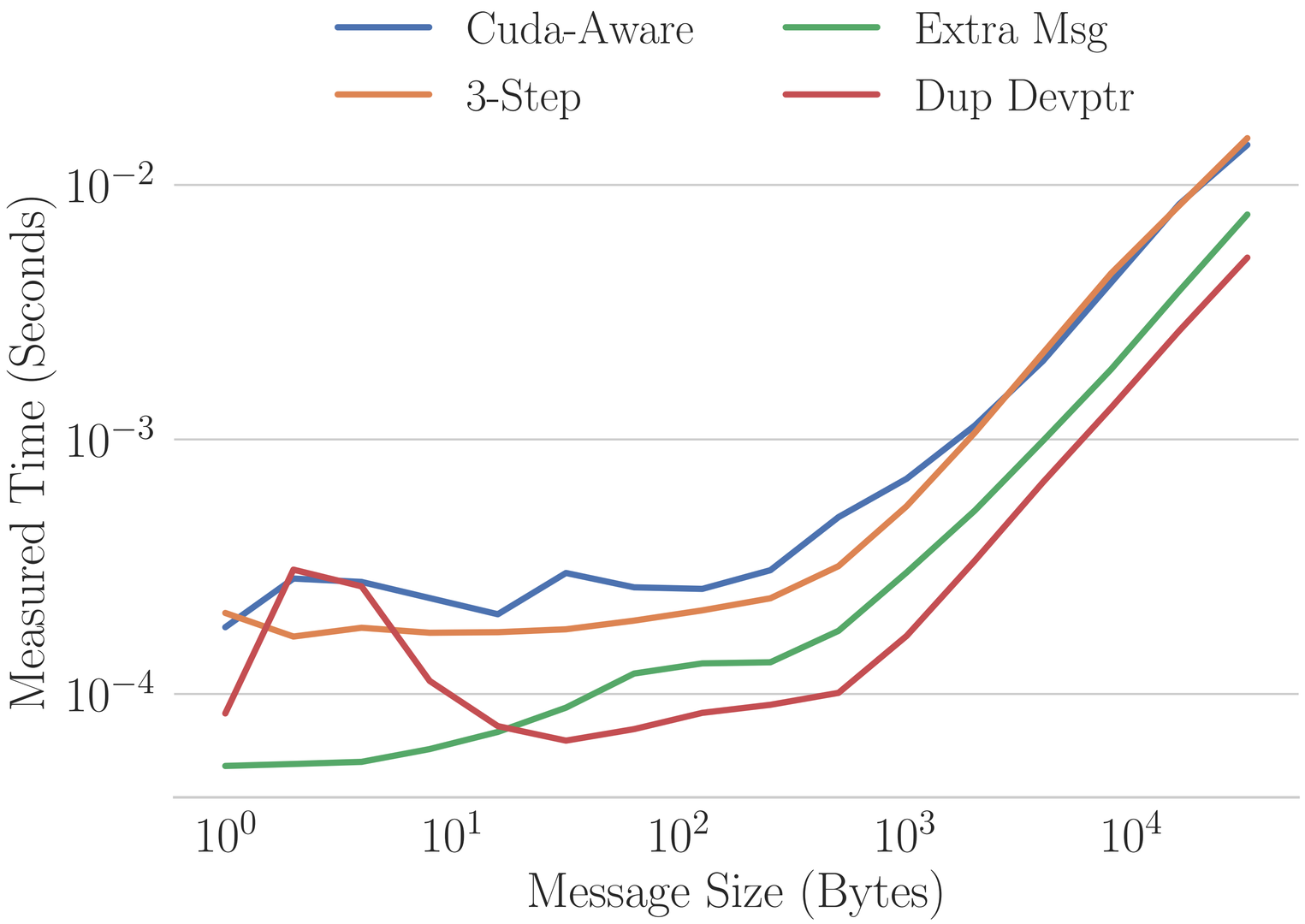
Summit : MPI_Alltoall



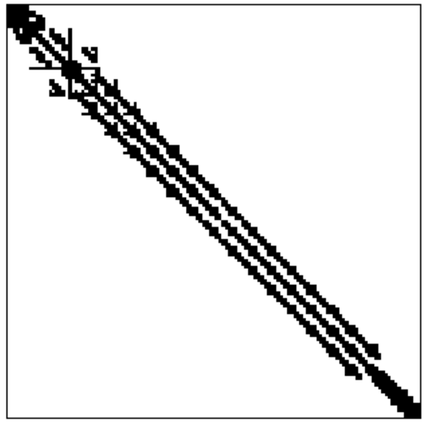
Lassen : MPI_Alltoallv

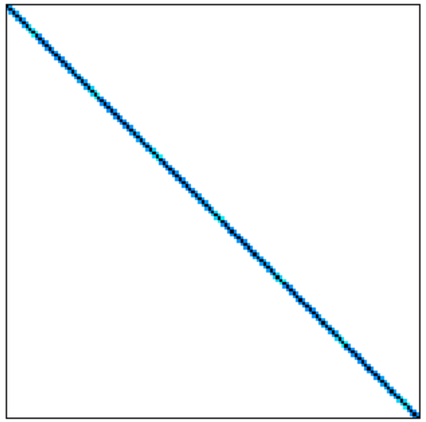


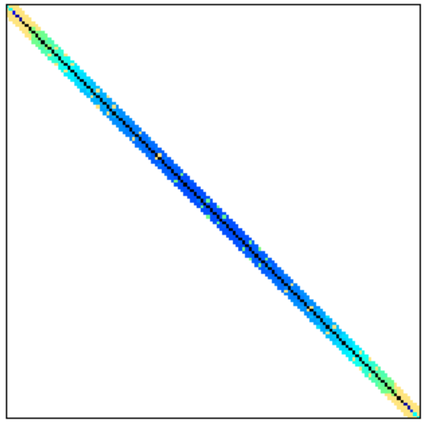
Summit : MPI_Alltoallv

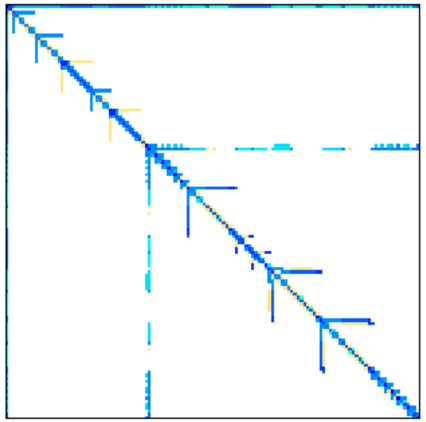


MPI_Neighbor_alltoallv

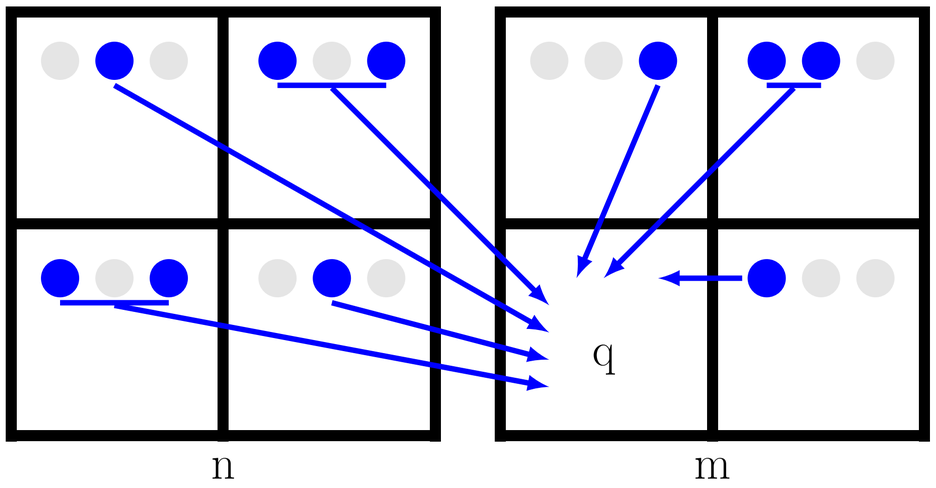




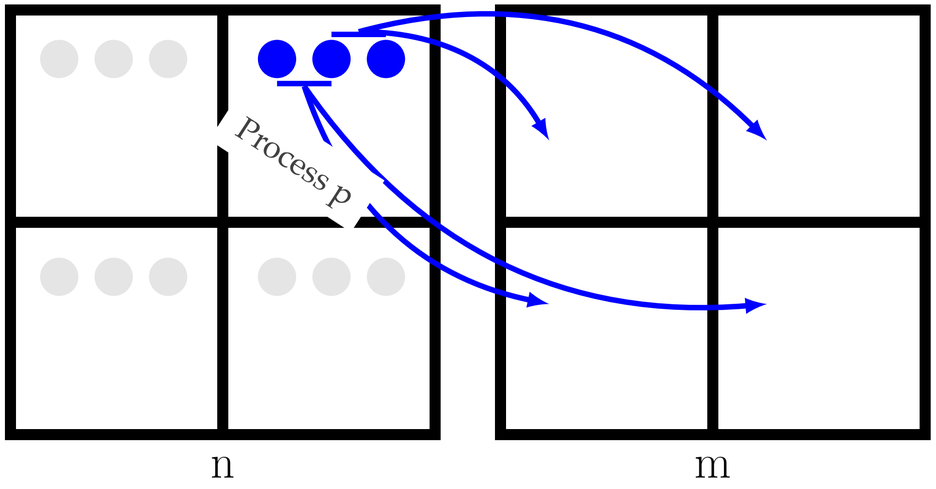




Standard Neighbor Communication

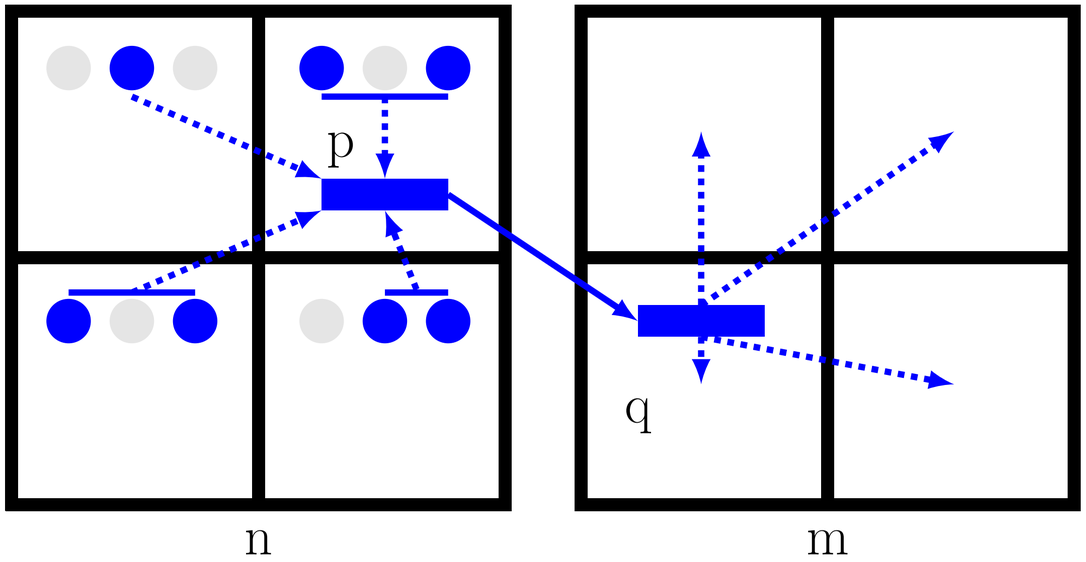


Multiple non-local messages

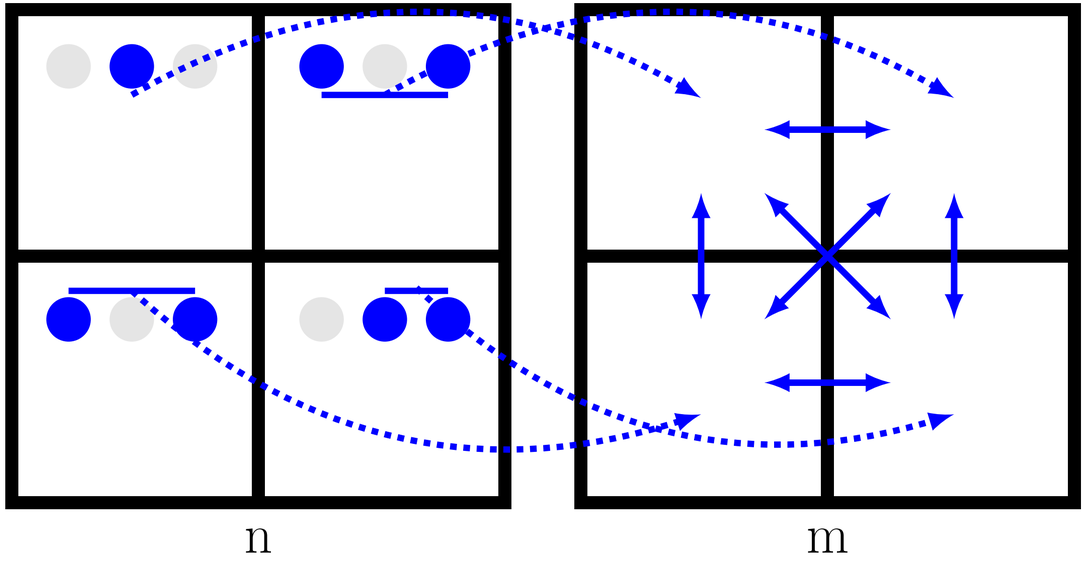


Multiple non-local messages and redundant data

Aggregated Neighbor Communication

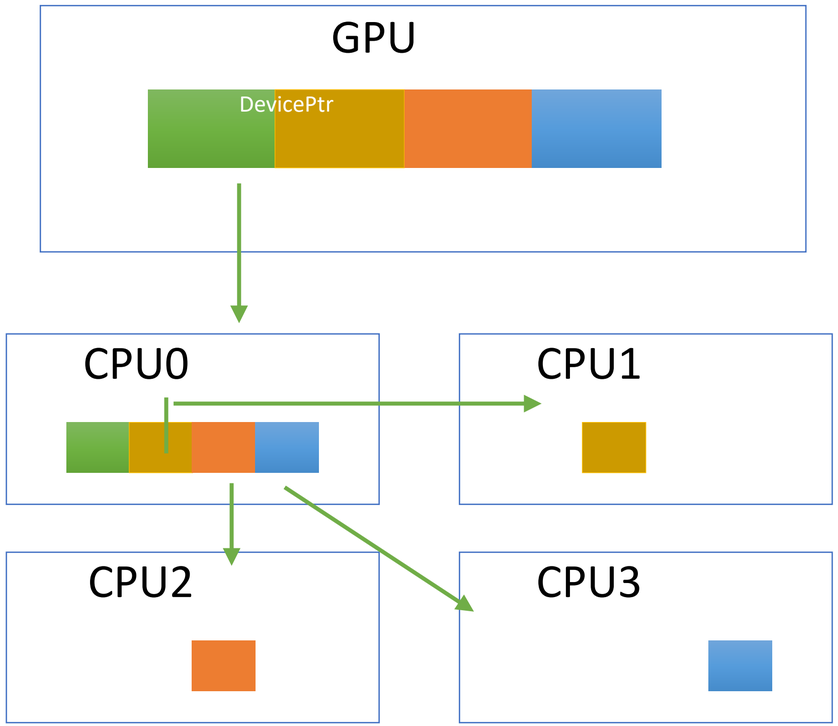


3-Step Aggregation

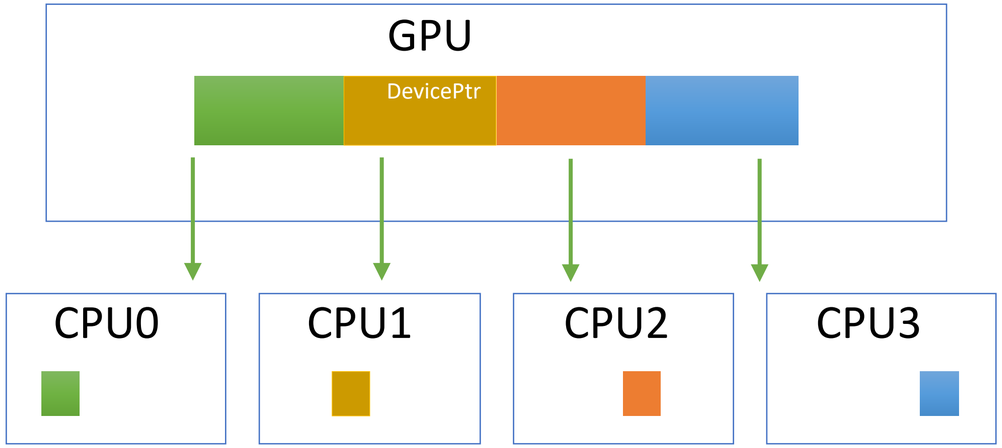


2-Step Aggregation

Using Multiple CPU Cores Per GPU

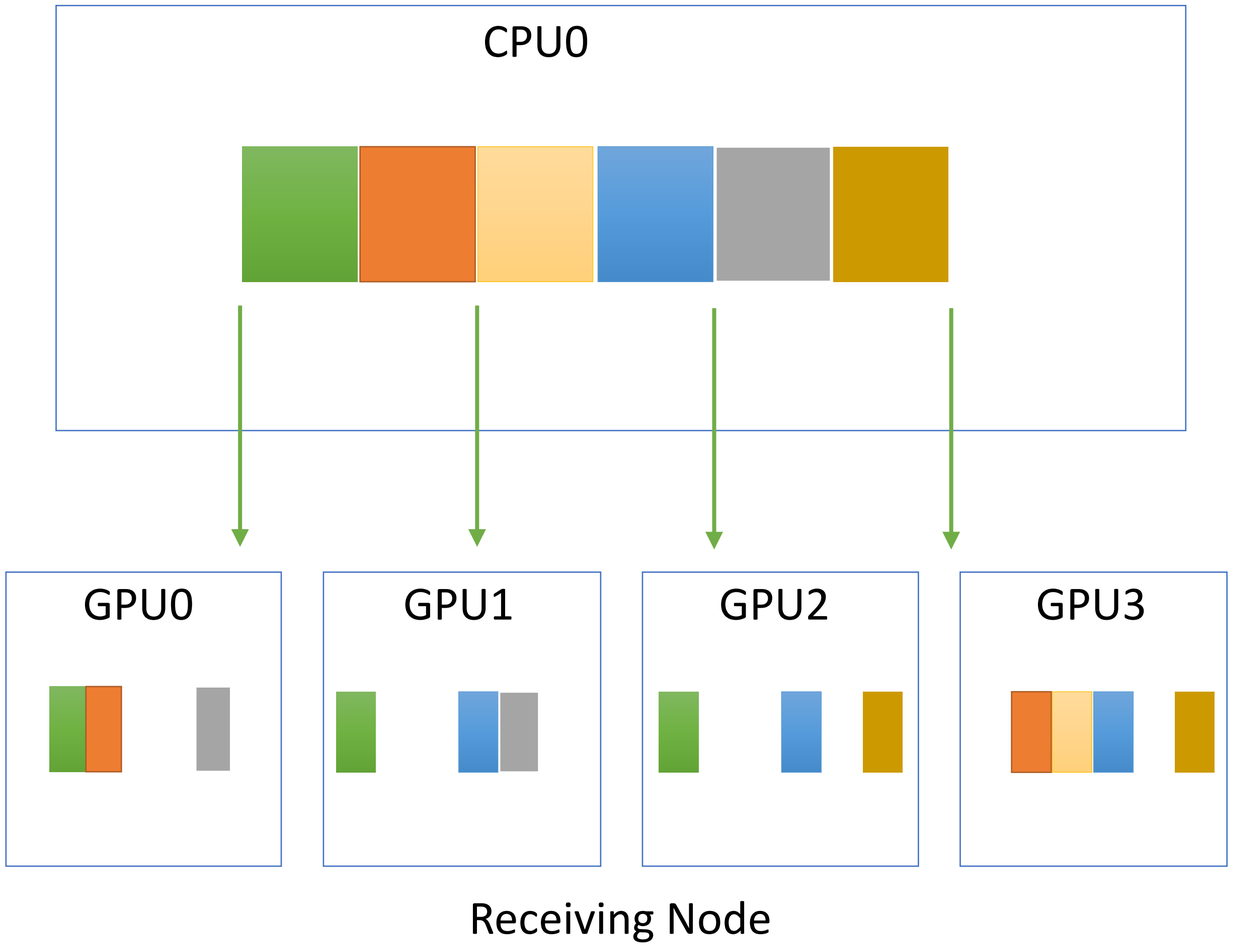


Extra Message Approach



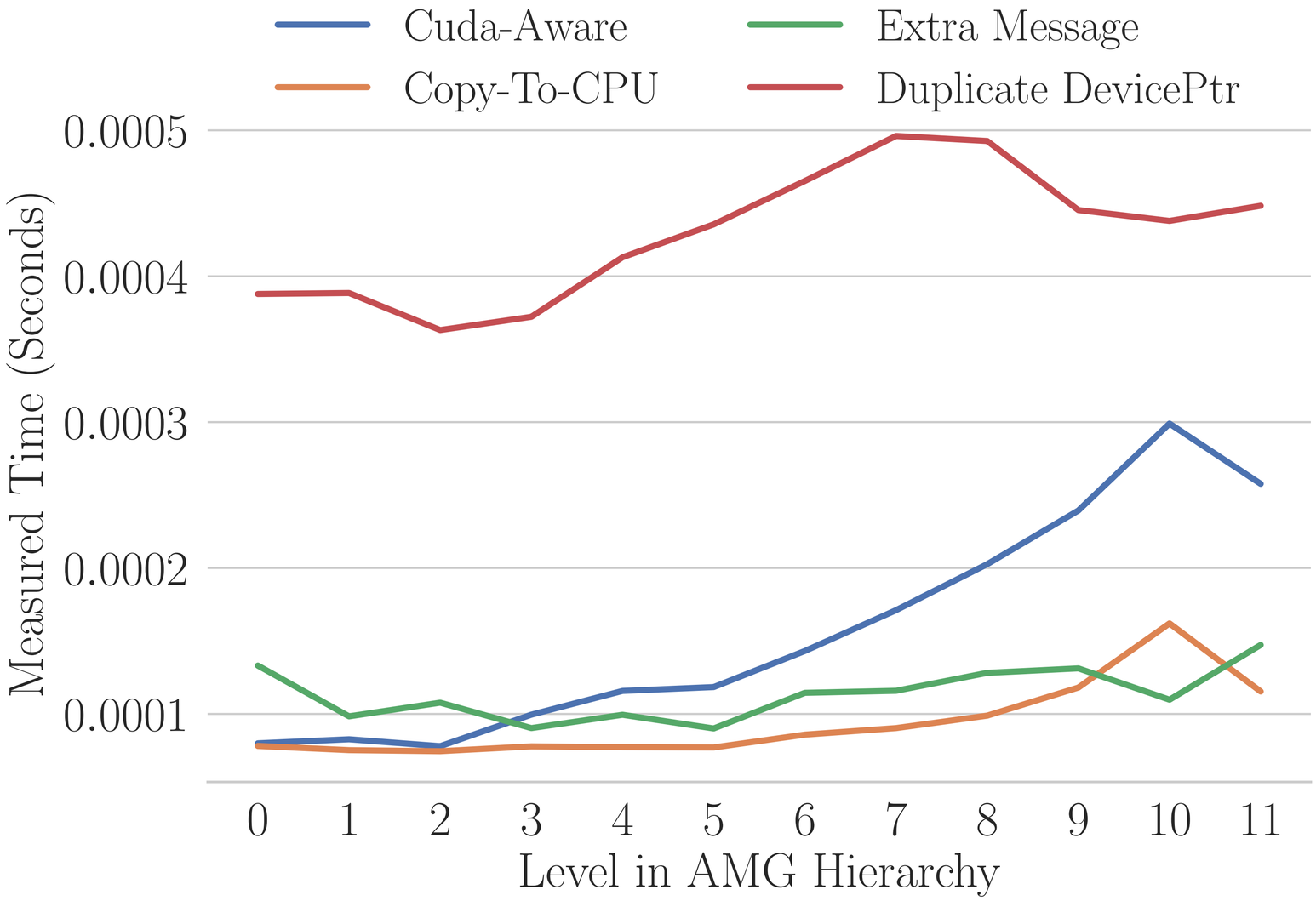
Duplicate Device Pointer Approach

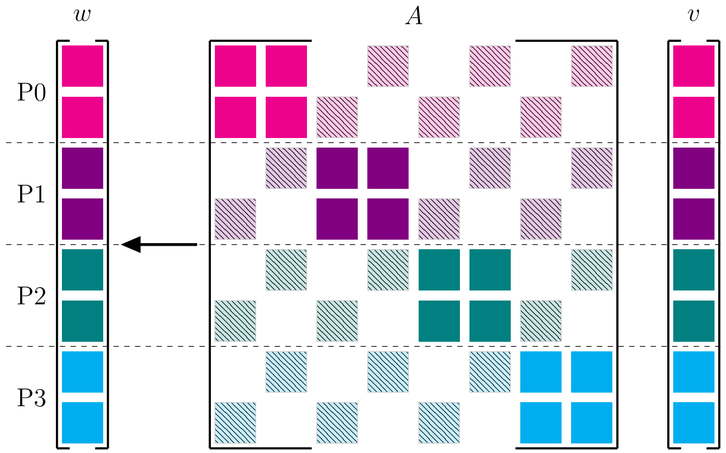
Locality-Aware Aggregation : Current Issues



- **Need to re-pack data on receiving node to remove duplicate values!**

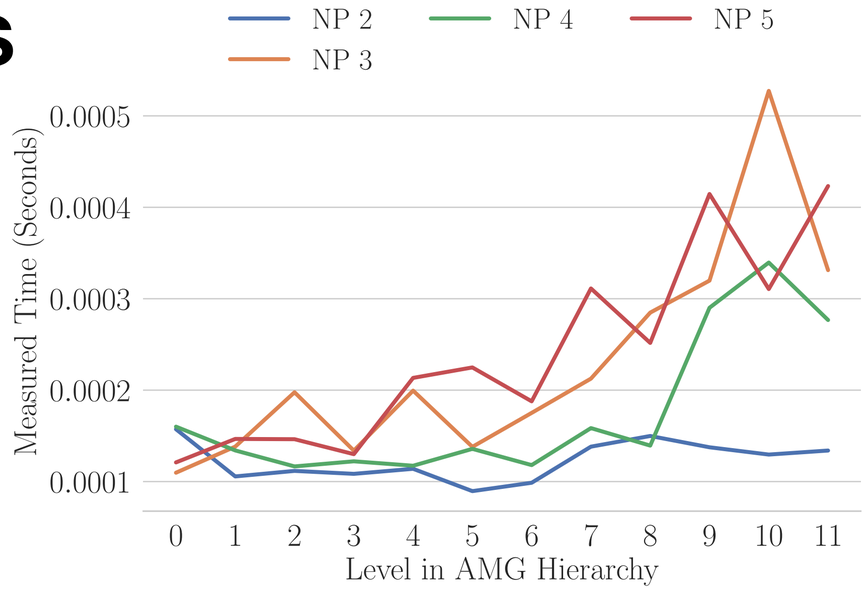
Locality-Aware Sparse Matrix-Vector Multiply



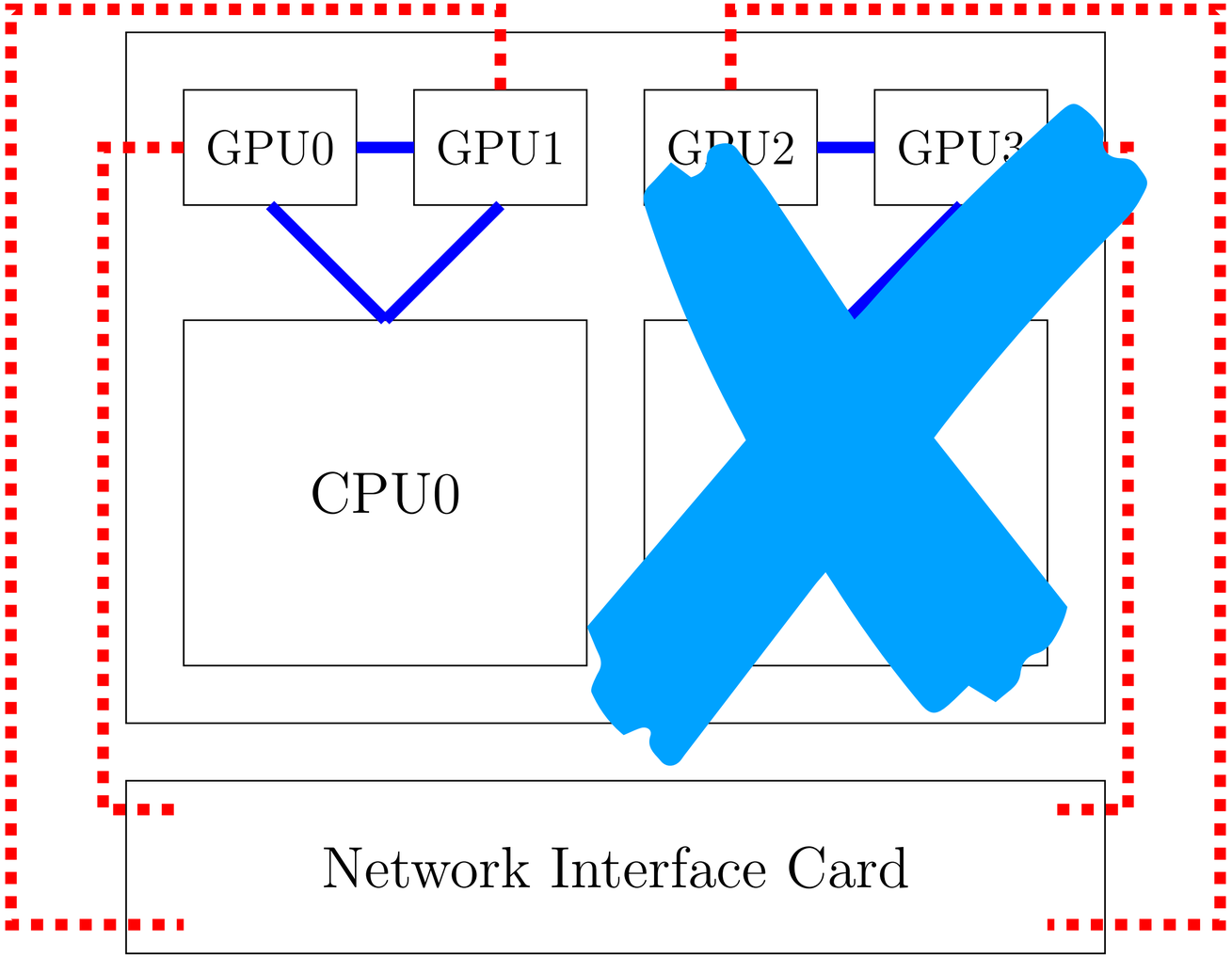


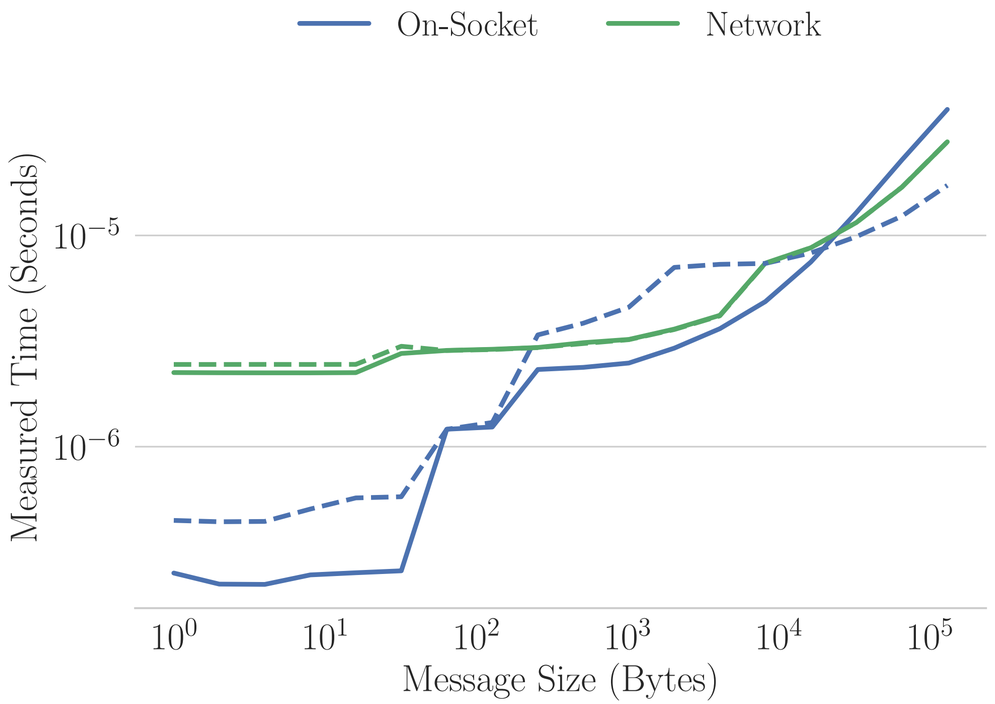
Locality-Aware Neighborhood Collective : Open Questions

- How many CPU cores should be used?
- Should data be aggregated? If so, on-socket? On-node?
- What is the best way to remove duplicate values?



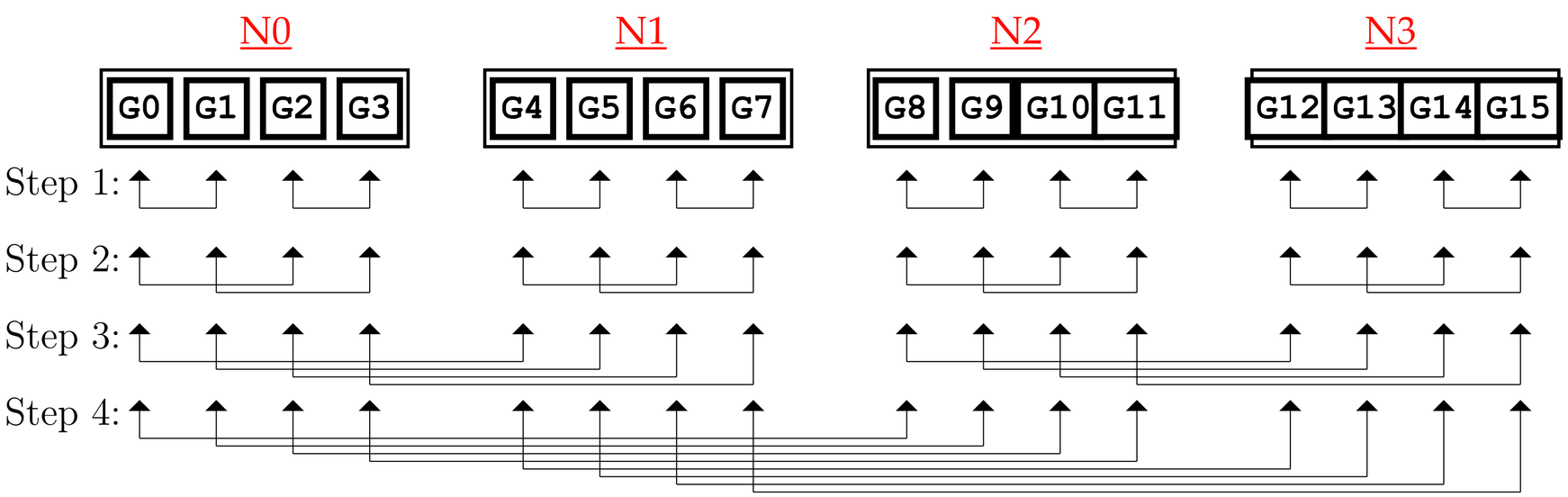
Pre-Frontier Collective Performance

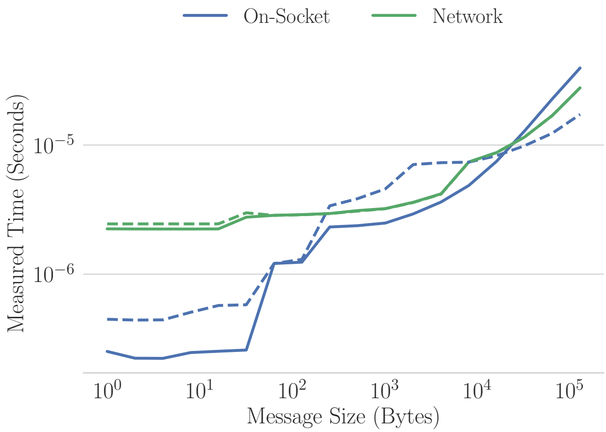




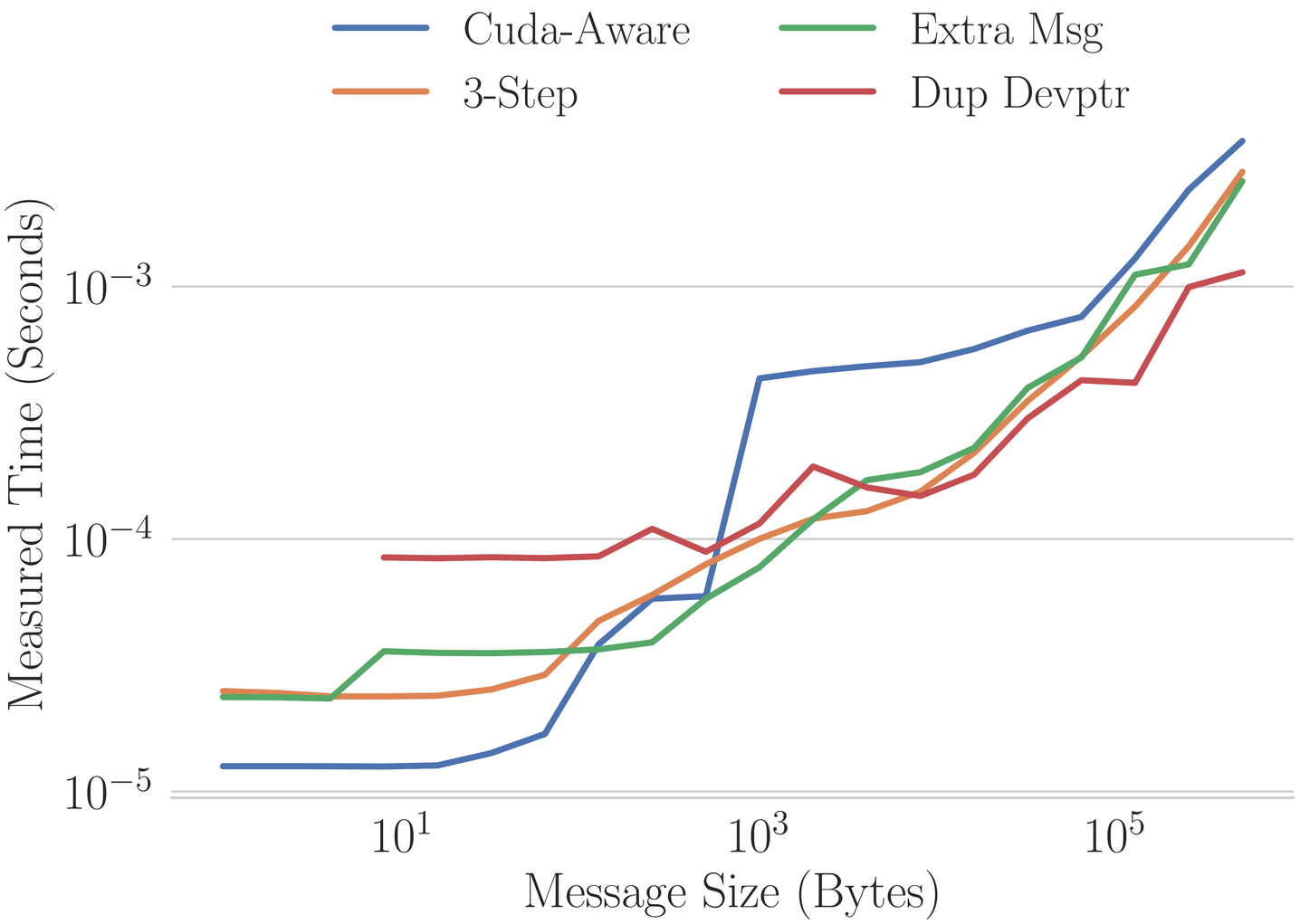
Reminder : Pre-Frontier Performance

Recursive Doubling

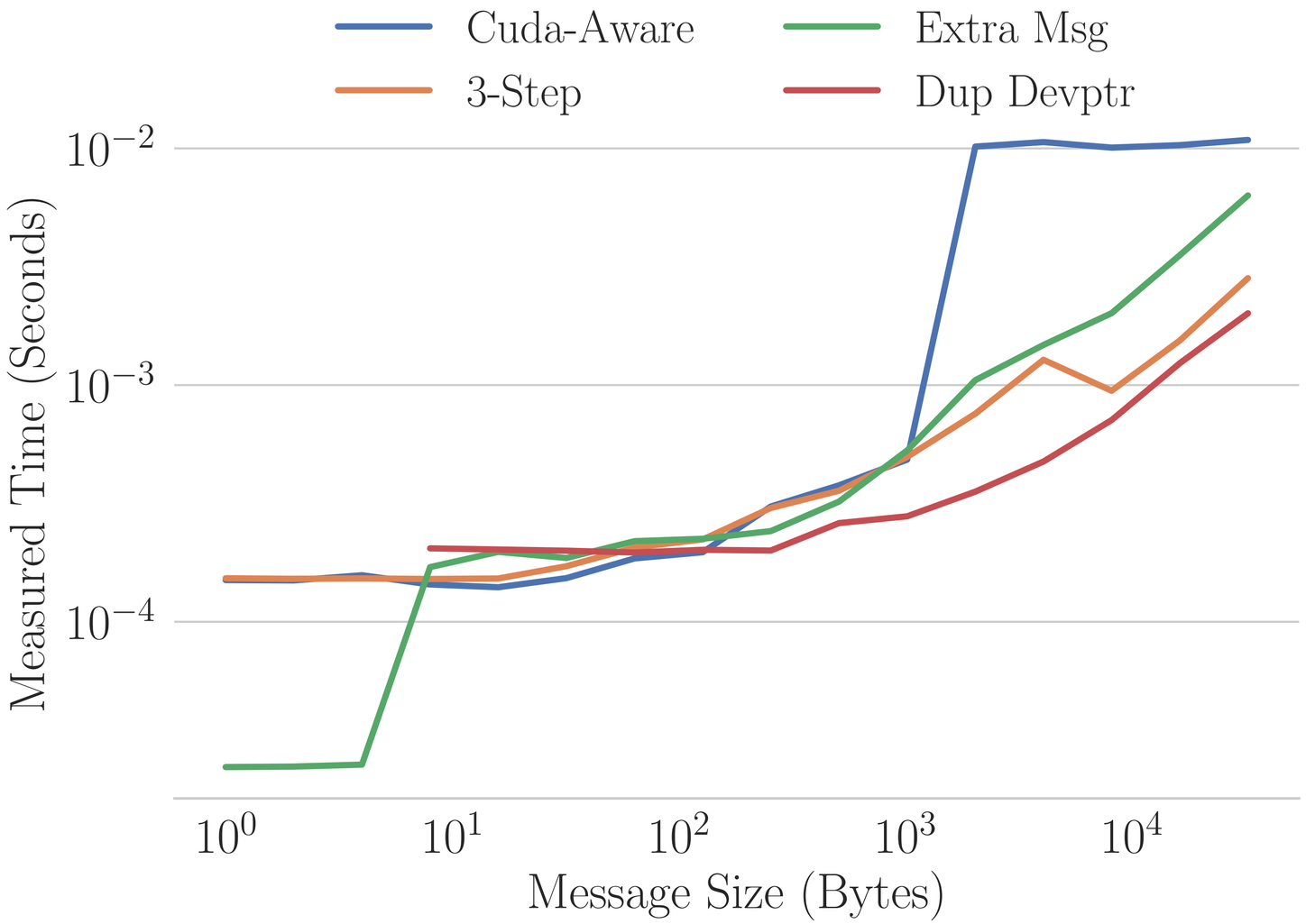




pre-Frontier : MPI_Allreduce



pre-Frontier : MPI_Alltoall



pre-Frontier : MPI_Alltoallv