2022年4月 更新版

Type IIサブシステム向けMPI環境の整備について

- 「不老」Type IIサブシステムで利用できるMPIとその性能について、情報を提供します。参考にしてください。

「不老」Type IIで利用できるGPU+MPI関係のmodule構成例

- 利用できる構成（の例）
  - 最新のCUDA (CUDA 11.6.2)の基本設定（MPIもNCCLも使える）
    - `module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7`
  - UCXを明示的にload
    - `module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7 ucx_cuda/1.11.2`
  - HPCXを明示的にload
    - `module load cuda/11.6.2 nccl/2.12.7 hpcx/2.9.0`
  - HPCXをloadするとOpenMPIもHPCX側のものになるので、それを回避する
    - `module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7 hpcx/2.9.0_stack`
  - HPC SDKの基本設定（MPIもNCCLも使える）
    - `module load hpc_sdk/22.2`
  - UCXやHPCXを明示的にload（ucx_cudaをloadするにはcudaのloadも必要）
    - `module load hpc_sdk/22.2 hpcx/2.9.0`
    - `module load hpc_sdk/22.2 hpcx/2.9.0_stack`
    - `module load cuda/11.6.2 hpc_sdk/22.2 ucx_cuda/1.11.2`
    - （および、これらの組み合わせ）

基本的にはこれらを利用すればOKです。特定の機能を活用したい場合はその他のmoduleも活用してください。

- HPC-X：MPIを高速化するツール群
- UCX： 片方向通信の高速化に有効（HPC-Xに含まれる）

HPC-X(hpcx) moduleの効果について

- 自作の通信性能ベンチマークとOSU Micro-Benchmarkdsで色々と測定してみたが、hpcxやhpcx_stackのmoduleをloadすることで性能が向上するパターンは見つからなかった。
  - hpcxをloadするとncclが遅くなるケースがあったため注意
  - 具体例は終盤のグラフにて

UCXの効果について

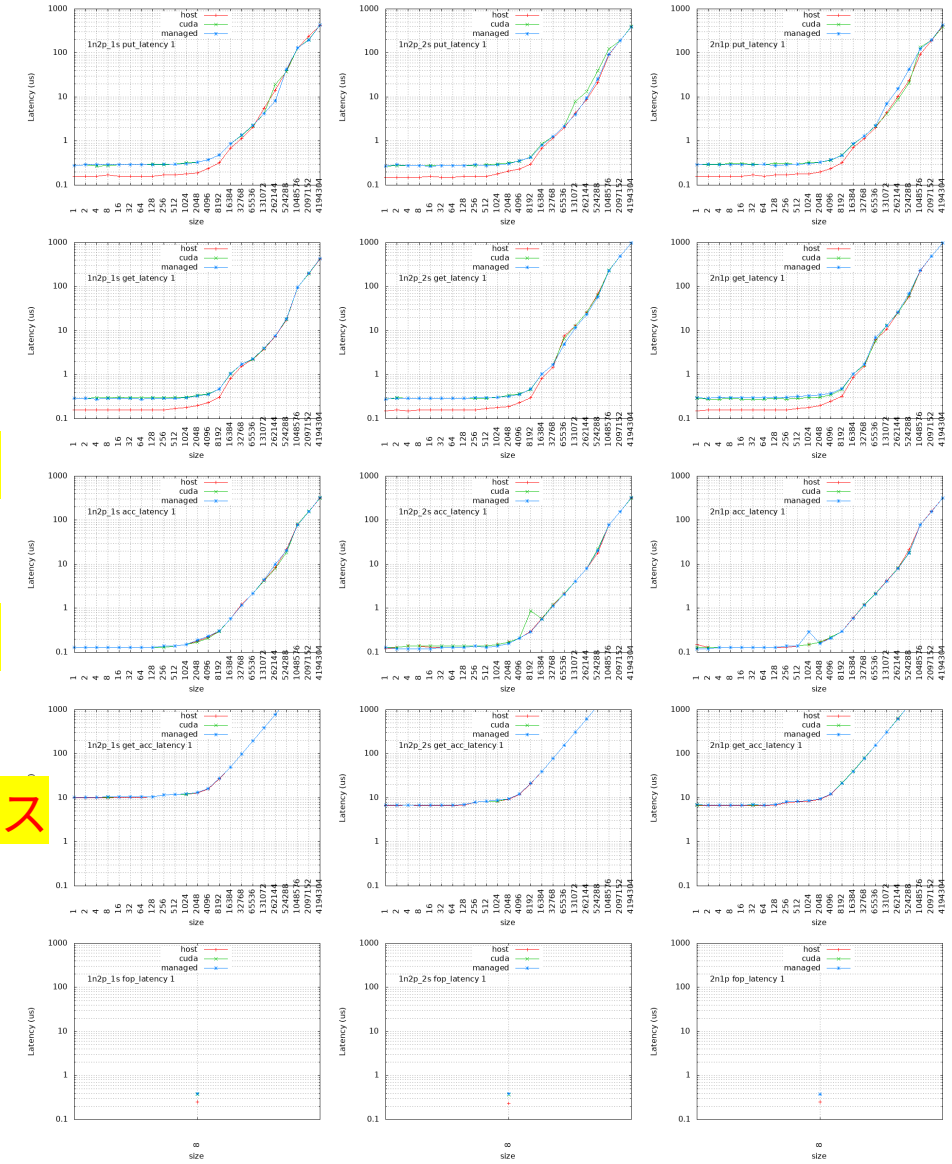
- mpirunに--mca pml ucx -mca osc ucxオプションを追加することでUCXの機能を有効化できるとされており、one-sided通信（片方向通信、put/get）の性能向上が期待できる。

- OSU Micro-Benchmarkdsのone-sided通信性能を比較してみた

- 比較結果（次ページ以降のグラフを参照）
  - 確かにオプションを付けると性能に影響が出る
  - ucx_cudaやhpcx, hpcx_stackのloadは特に影響しなかった
  - 利用するone-sided関数によって性能が上がったり下がったりするようなので注意が必要
    - 基本的なput/getの性能は向上するようだ

グラフの見方

- 左から順に
- 1ノード、2プロセス
- （1ソケット内）
- 1ノード、2プロセス
- （2ソケット間）
- 2ノード、各1プロセス
- （ノード間）



put

get

acc
(Accumulate with Active/Passive Synchronization)

get_acc
(Get_accumulate with Active/Passive Synchronization)

fop
(Fetch and Op with Active/Passive Synchronization)

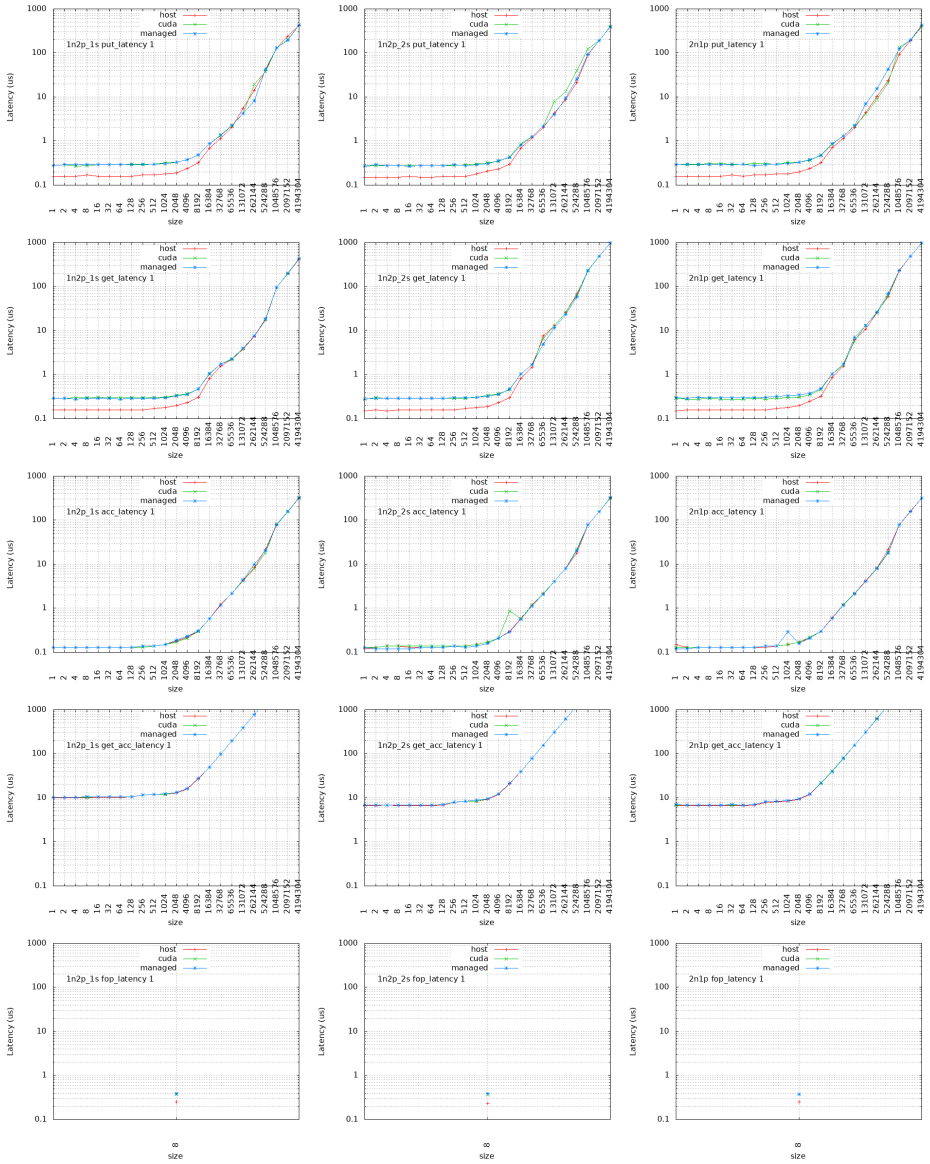
通信パターンが違う

- mpirunにめぼしいオプションは付けていない版。

(os0) module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7

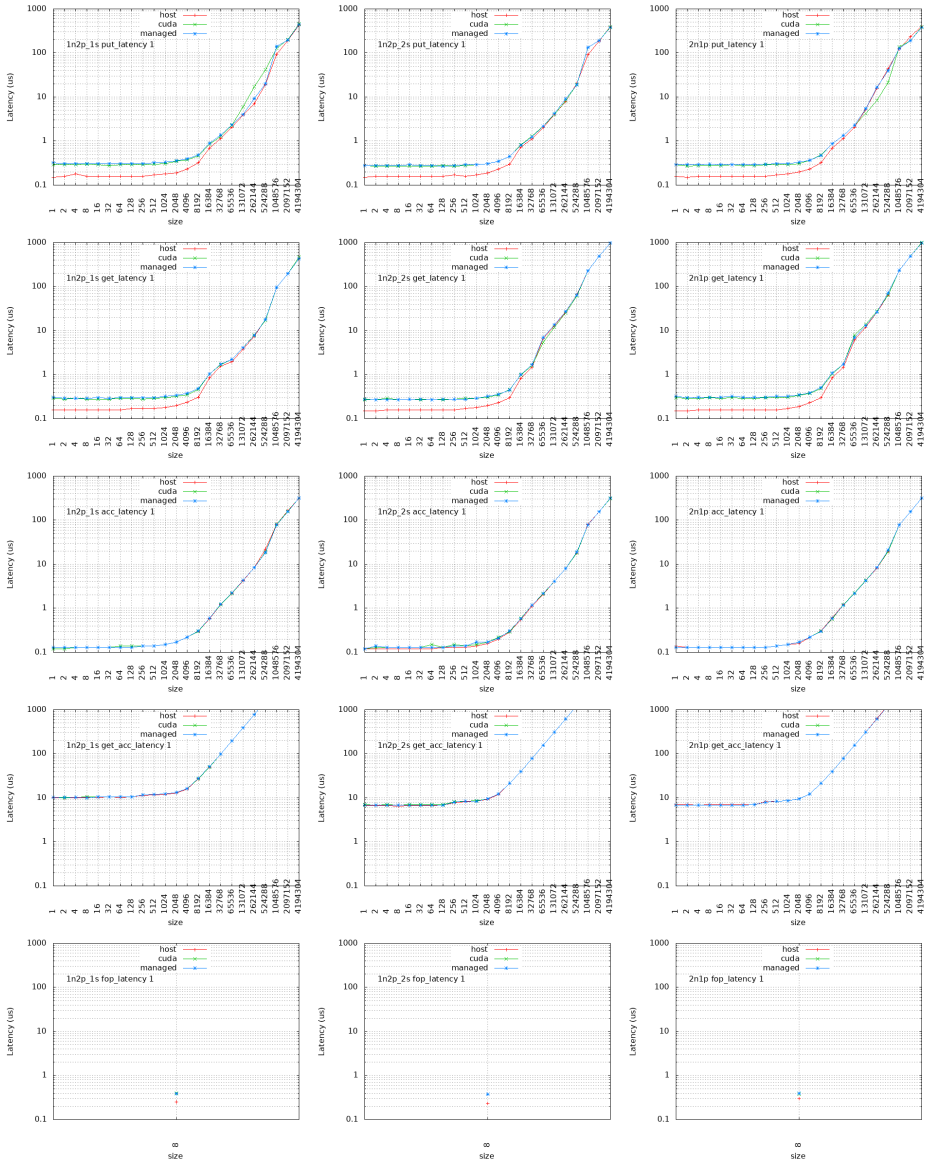
以降のグラフでは
この性能を基準
として比較する。

latencyなので
低いほど良い。



- ucx_cudaをload。 mpirunへのオプション追加はなし。
- ucx_cudaなしと変わらぬ性能。

(os6) module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7 ucx_cuda/1.11.2



- ucx_cudaをload。 mpirunに--mca pml ucx -mca osc ucxを追加。
- 通信パターンによって性能が変化。これがUCXの効果か？

(os7) module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7 ucx_cuda/1.11.2

基準と比べて……

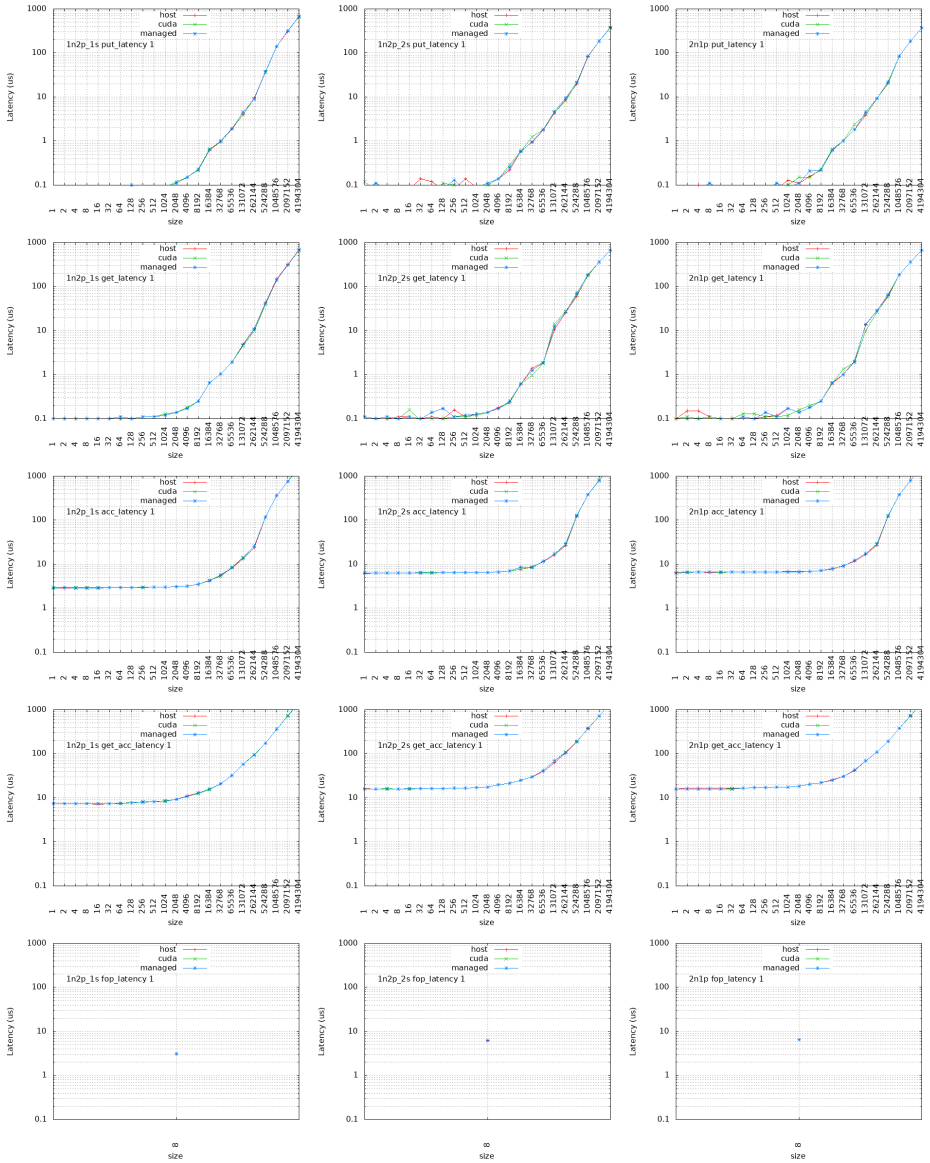
高速化

高速化

低速化

どちらとも言えない

低速化



- hpcxをload。mpirunへのオプション追加はなし。
- get_acc_latencyだけ高速化。他はほんのわずかに低速化（ほぼ同じ、誤差？）。

(os3) module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7 hpcx/2.9.0

基準と比べて……

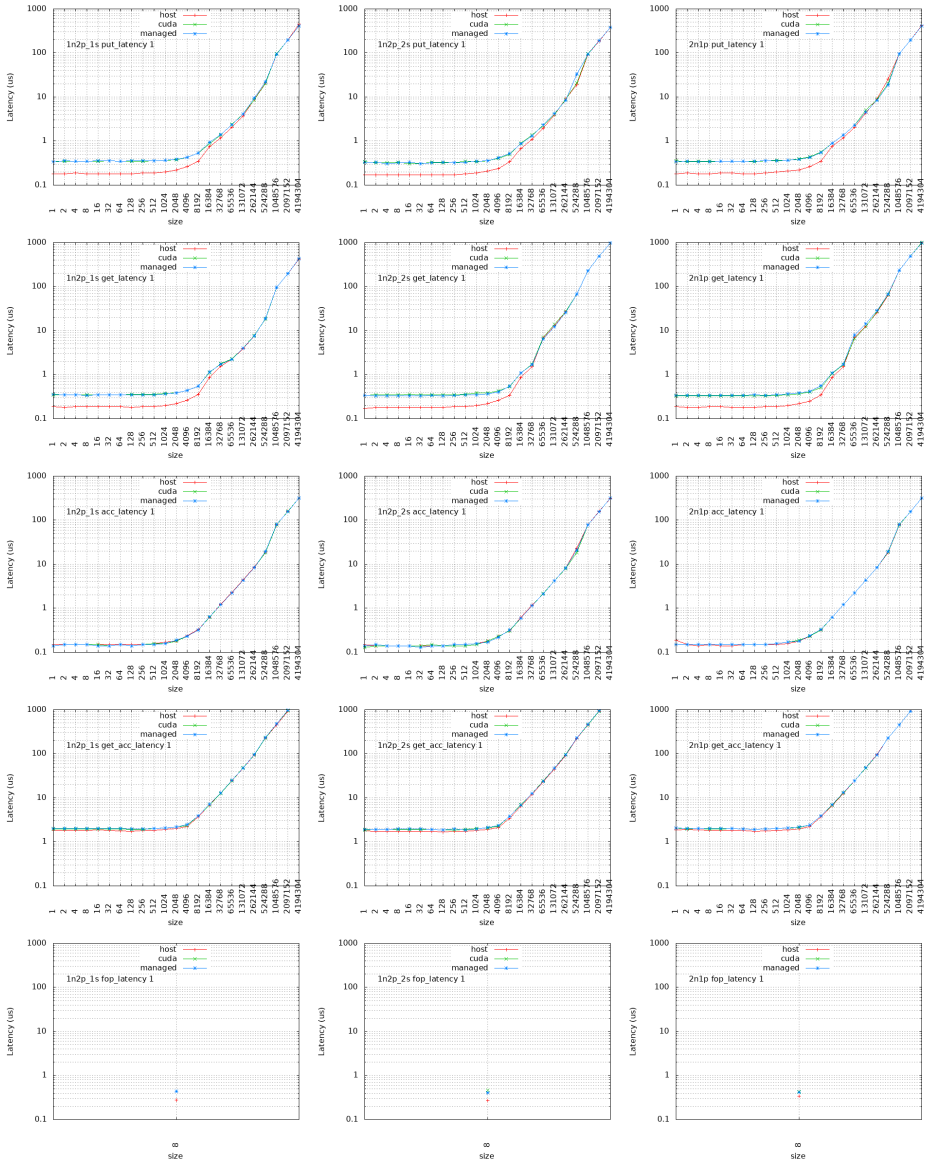
ほんの僅かに低速化

ほんの僅かに低速化

ほんの僅かに低速化

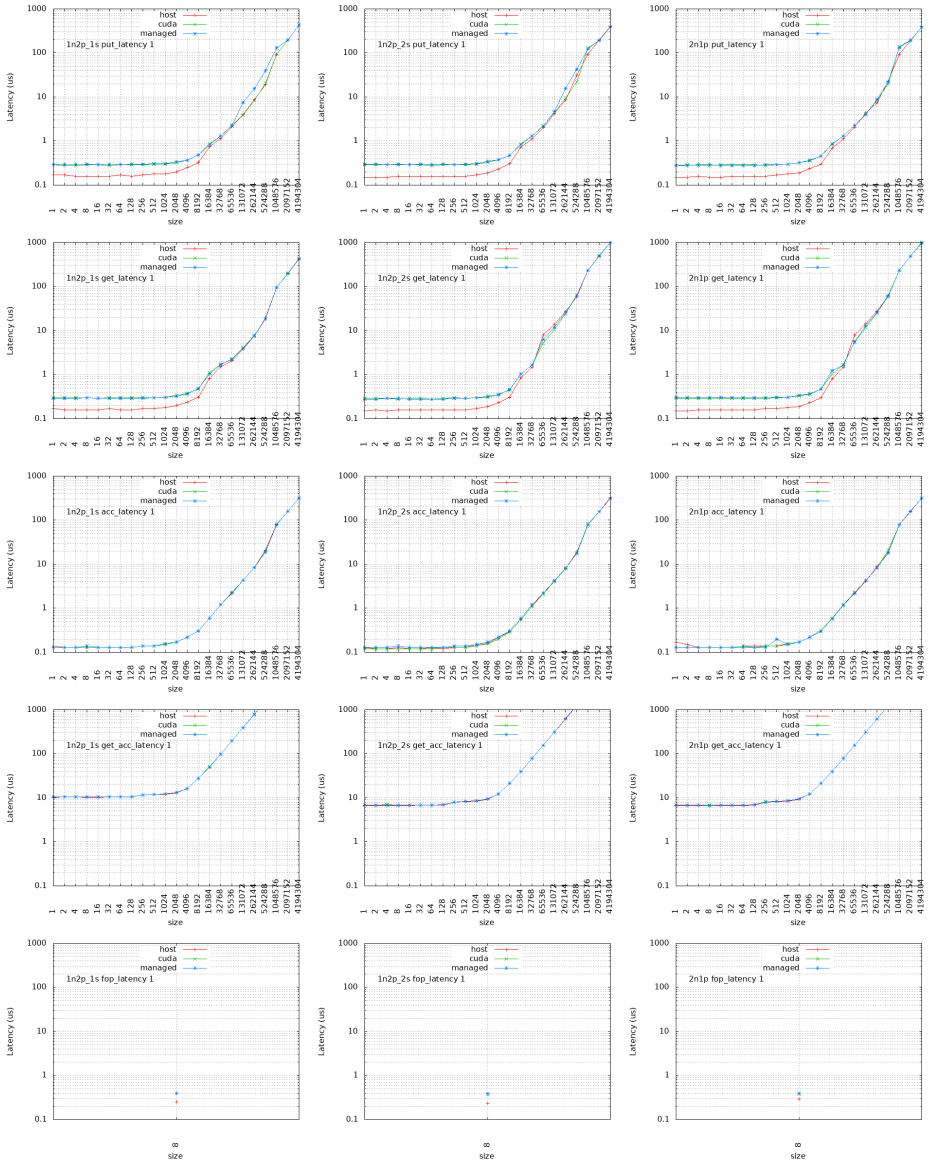
高速化

ほんの僅かに低速化



- hpcx_stackをload。mpirunへのオプション追加はなし。
- hpcx_stackなしと変わらぬ性能。

(os1) module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7 hpcx/2.9.0_stack



- mpirunに--mca pml ucx -mca osc ucxを追加。
- 通信パターンによって性能が変化。ucx_cudaを追加しなくても同様の効果が得られる模様。

(os2) module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7

基準と比べて……

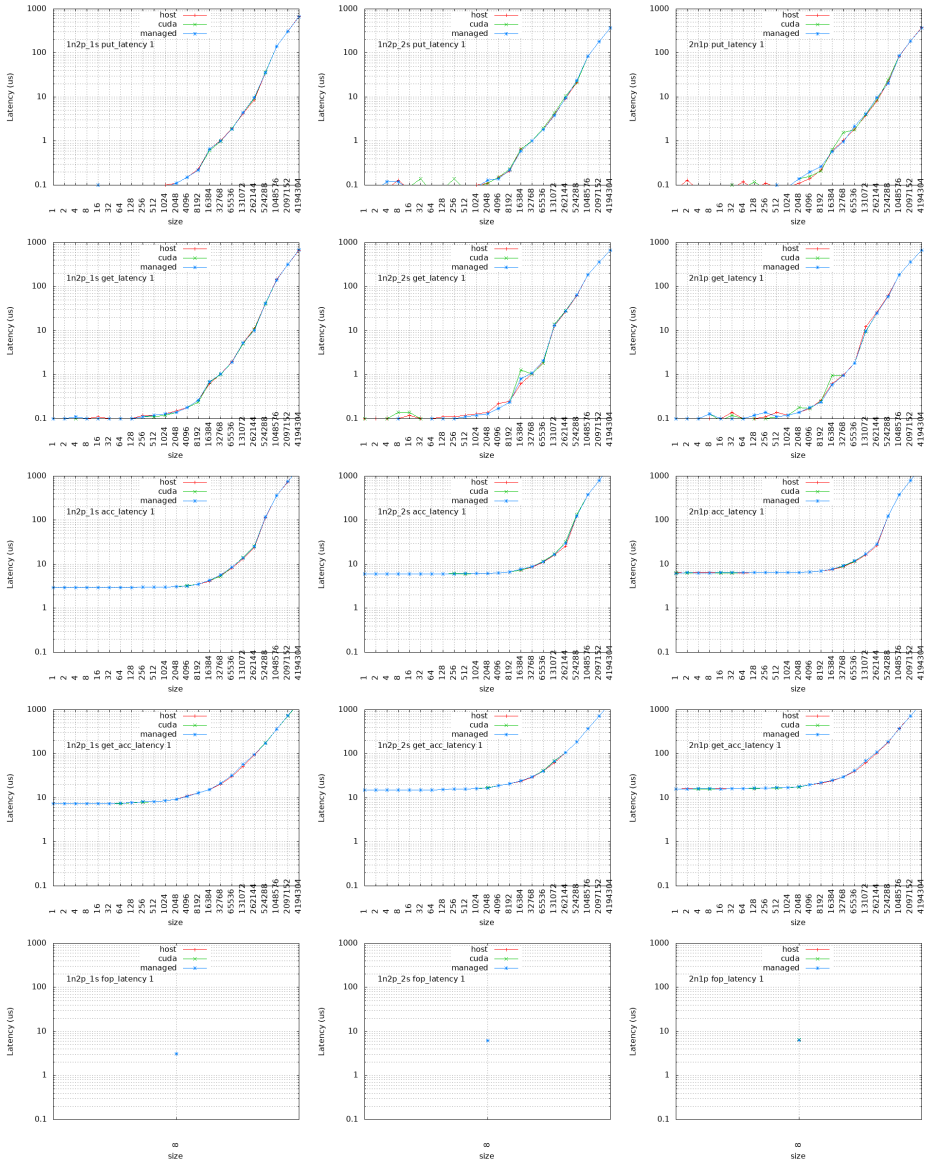
高速化

高速化

低速化

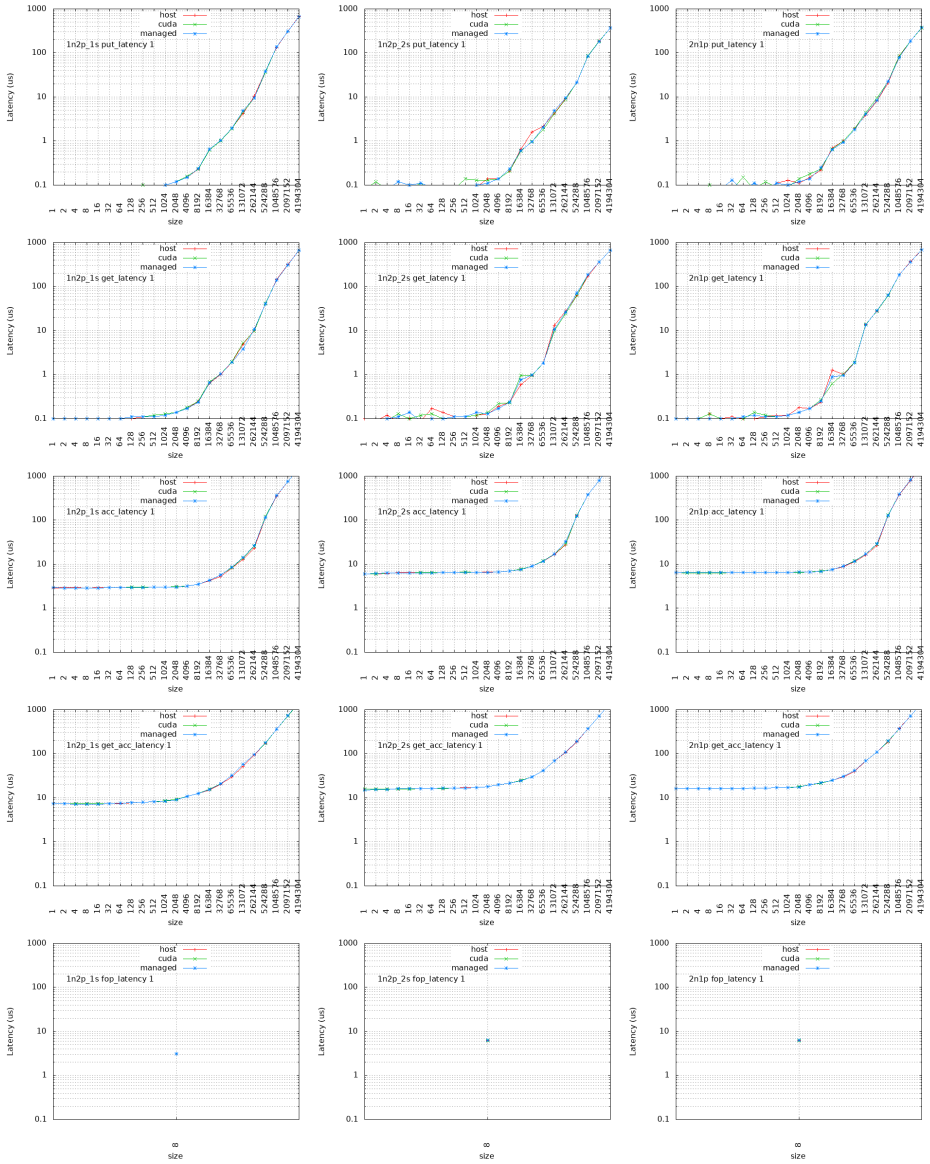
どちらとも言えない

低速化



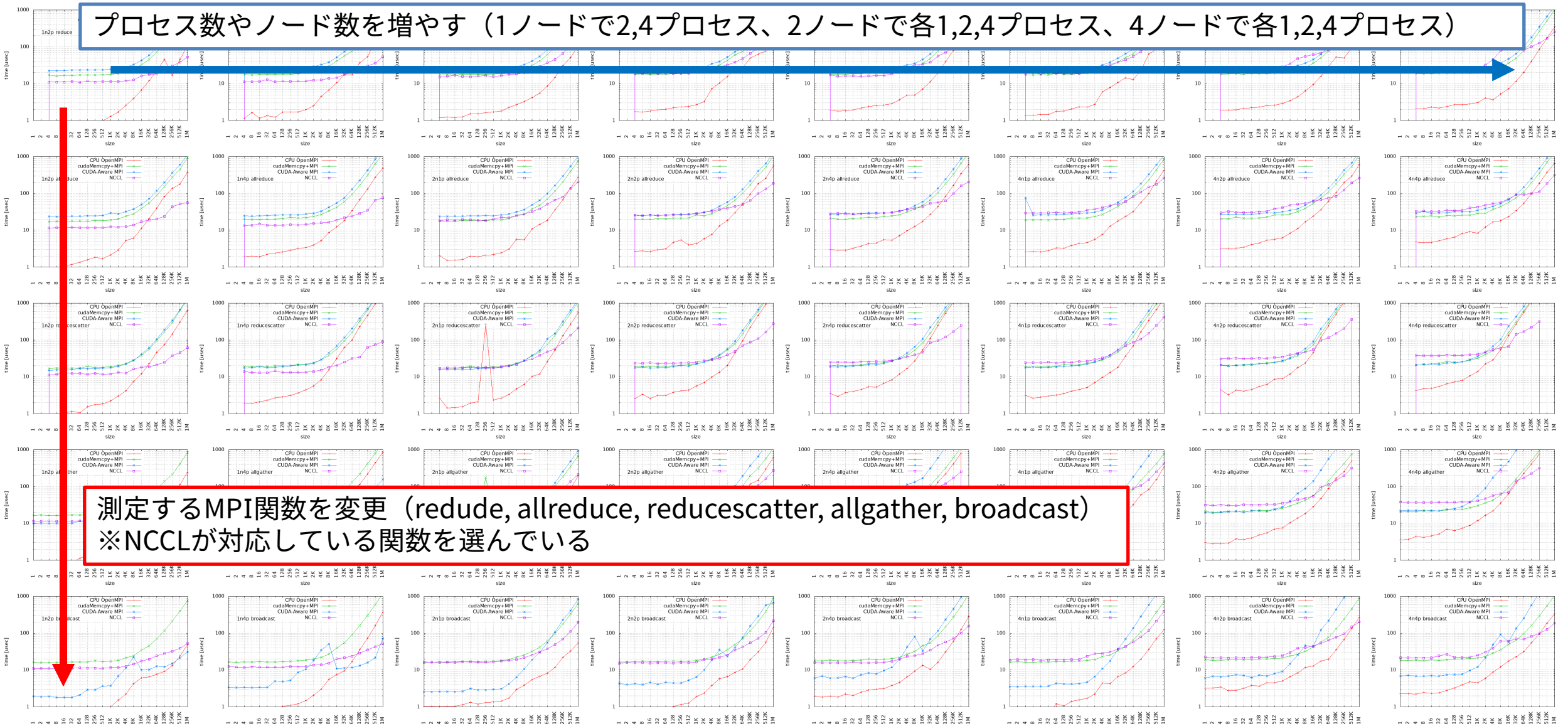
- mpirunに--mca pml ucx -mca osc ucxを追加。
- 前ページと変わらない。hpcx_stackのload有無は特に意味がないようだ。

(os5) module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7 hpcx/2.9.0_stack



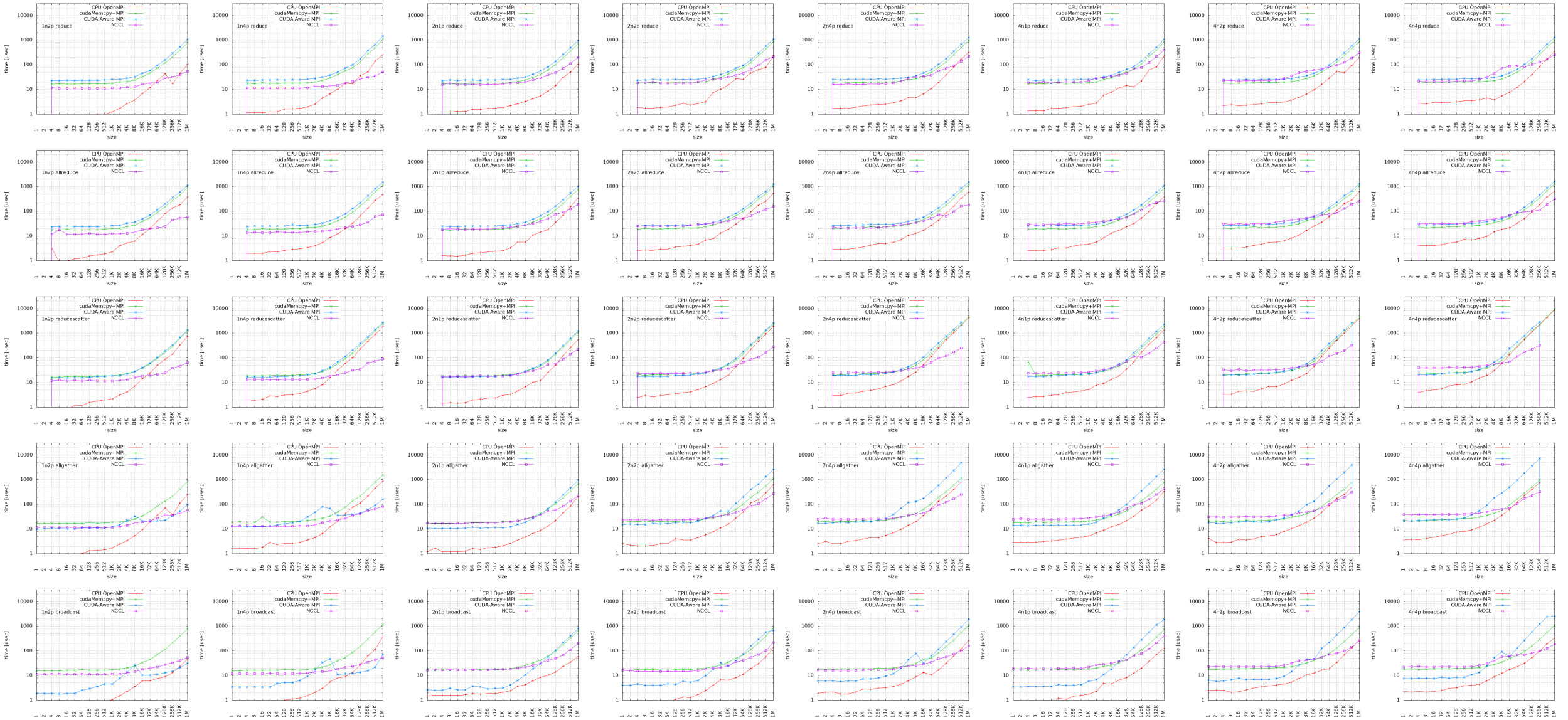
自作のベンチマークプログラムによる性能確認

グラフの見方


プロセス数やノード数を増やす（1ノードで2,4プロセス、2ノードで各1,2,4プロセス、4ノードで各1,2,4プロセス）
測定するMPI関数を変更（redude, allreduce, reducescatter, allgather, broadcast）
※NCCLが対応している関数を選んでいる

- 自作の単純な性能評価プログラムによる通信性能比較
- 一部CUDA-Aware MPIよりcudaMemcpy+MPIの方が速いものがある点と、NCCLが負けているものがある点に注意が必要そう

CUDA 11.6 (module load cuda/11.6.2 openmpi_cuda/4.1.2 nccl/2.12.7)



- 自作の単純な性能評価プログラムによる通信性能比較

hpc_sdk22.2 (module load hpc_sdk/22.2)

- 全体的に、update指示文は遅く、ncclは速い（CUDAと同等）
- directはupdate指示文を使わずhost_data use_device指示文を使ったもの。
  MPI + CUDA-Aware MPI並の性能が出ていないものもあり、改善の余地がある？