Implementation of Parallel 3-D FFT with 2-D Decomposition on GPU Clusters

Daisuke Takahashi

Center for Computational Sciences
University of Tsukuba, Japan

Outline

- Background
- Related Works
- Objectives
- 3-D FFT with 2-D Decomposition
- Performance Results
- Conclusion

Background (1/2)

- The fast Fourier transform (FFT) is an algorithm widely used today in science and engineering.
- Parallel 3-D FFT algorithms on distributed-memory parallel computers have been well studied.
- November 2025 TOP500 Supercomputing Sites
  - El Capitan: 1,809 PFlops (11,340,000 Cores)
  - Frontier: 1,353 PFlops (9,066,176 Cores)
  - Aurora: 1,012 PFlops (9,264,128 Cores)
- Recently, the number of cores keeps increasing.

Background (2/2)

- A typical decomposition for performing a parallel 3-D FFT is slabwise.
  - A 3-D array $x(N_1, N_2, N_3)$ is distributed along the third dimension N_3.
  - N_3 must be greater than or equal to the number of MPI processes.
- This becomes an issue with very large MPI process counts for a massively parallel cluster of GPUs.

Related Works

- P3DFFT [Pekurovsky 2012]
  - 3-D real-to-complex/complex-to-real FFT with 2-D decomposition
- 2DECOMP&FFT [Li and Laizet 2010]
  - 3-D complex-to-complex and real-to-complex/ complex-to-real FFT with 2-D decomposition
- PFFT [Pippig 2013]
  - 3-D complex-to-complex and real-to-complex/ complex-to-real FFT with 2-D decomposition
- heFFTe [Alaya et al. 2020]
  - 3-D real-to-complex/complex-to-real FFT with 2-D decomposition on GPU clusters

Objectives

- Implementation and evaluation of highly scalable 3-D FFT with 2-D decomposition on GPU clusters.
- Reduce the communication time for larger numbers of MPI processes.
- A comparison between 1-D and 2-D decomposition for 3-D FFT.

3-D DFT

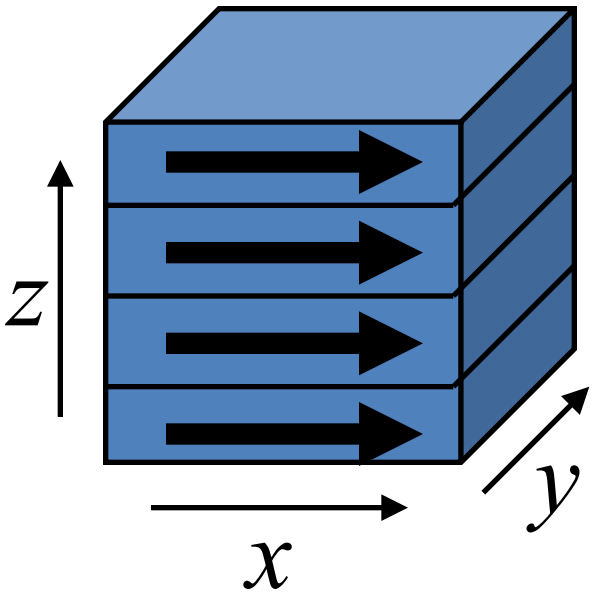
- 3-D discrete Fourier transform (DFT) is given by

$$y(k_1, k_2, k_3) = \sum_{j_1=0}^{n_1-1} \sum_{j_2=0}^{n_2-1} \sum_{j_3=0}^{n_3-1} x(j_1, j_2, j_3)\, \omega_{n_3}^{j_3 k_3} \omega_{n_2}^{j_2 k_2} \omega_{n_1}^{j_1 k_1},$$

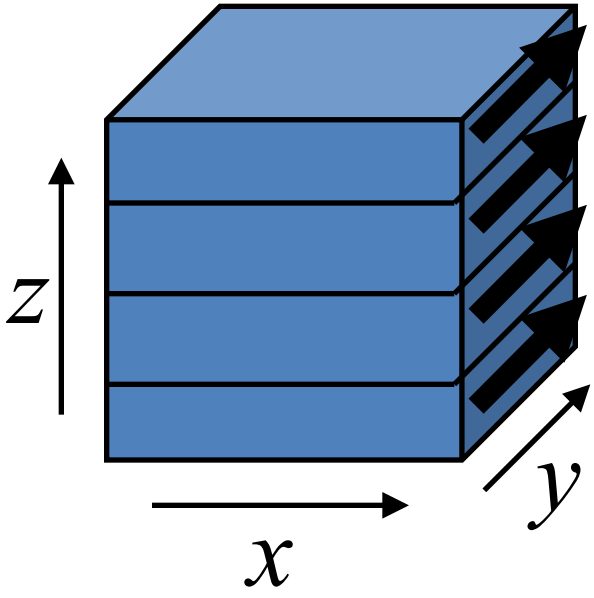
$$0 \le k_r \le n_r - 1, \omega_{n_r} = e^{-2\pi i / n_r}, 1 \le r \le 3$$

1-D Decomposition along the z-axis

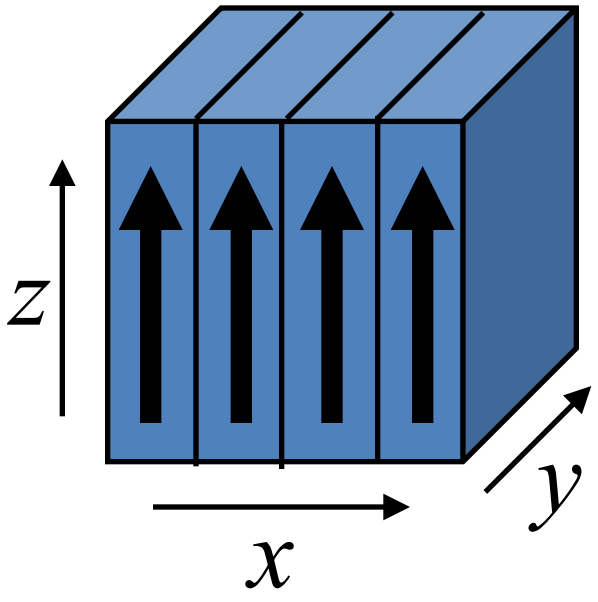
1. FFTs in x-axis



2. FFTs in y-axis



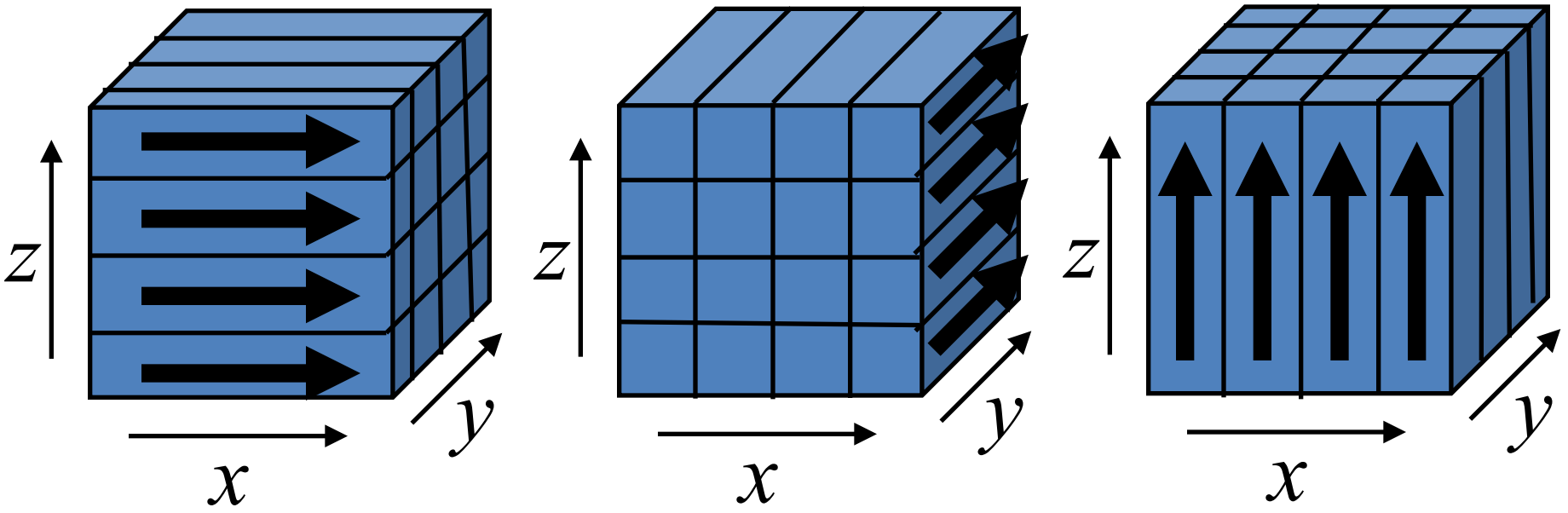
3. FFTs in z-axis



With a slab decomposition

2-D Decomposition along the y- and z-axes

1. FFTs in x-axis 2. FFTs in y-axis 3. FFTs in z-axis



With a pencil decomposition

Communication Time of 1-D Decomposition

- Let us assume for $N = N_1 \times N_2 \times N_3$-point FFT:
  - Latency of communication: L (sec)
  - Bandwidth: W (byte/sec)
  - The number of MPI processes: $P \times Q$
- One all-to-all communication among $P \times Q$ MPI processes
- Communication time of 1-D decomposition

$$T_{1\mathrm{dim}} \approx (PQ-1)\left(L + \frac{16N}{(PQ)^2 \cdot W}\right)$$

$$\approx PQ \cdot L + \frac{16N}{PQ \cdot W}\ (\mathrm{sec})$$

Communication Time of 2-D Decomposition

- Q simultaneous all-to-all communications among P MPI processes in the y-axis.
- P simultaneous all-to-all communications among Q MPI processes in the z-axis.
- Communication time of 2-D decomposition

$$T_{2\mathrm{dim}} \approx (P-1)\left(L + \frac{16N}{P^2Q \cdot W}\right) + (Q-1)\left(L + \frac{16N}{PQ^2 \cdot W}\right)$$

$$\approx (P+Q) \cdot L + \frac{32N}{PQ \cdot W}\ (\mathrm{sec})$$

Comparing Communication Time

- Communication time of 1-D decomposition

$$T_{1\mathrm{dim}} \approx PQ \cdot L + \frac{16N}{PQ \cdot W} \ (\mathrm{sec})$$

- Communication time of 2-D decomposition

$$T_{2\mathrm{dim}} \approx (P + Q) \cdot L + \frac{32N}{PQ \cdot W} \ (\mathrm{sec})$$

- By comparing two equations, the communication time of the 2-D decomposition is less than that of the 1-D decomposition for larger number of MPI processes $P \times Q$ and latency L.

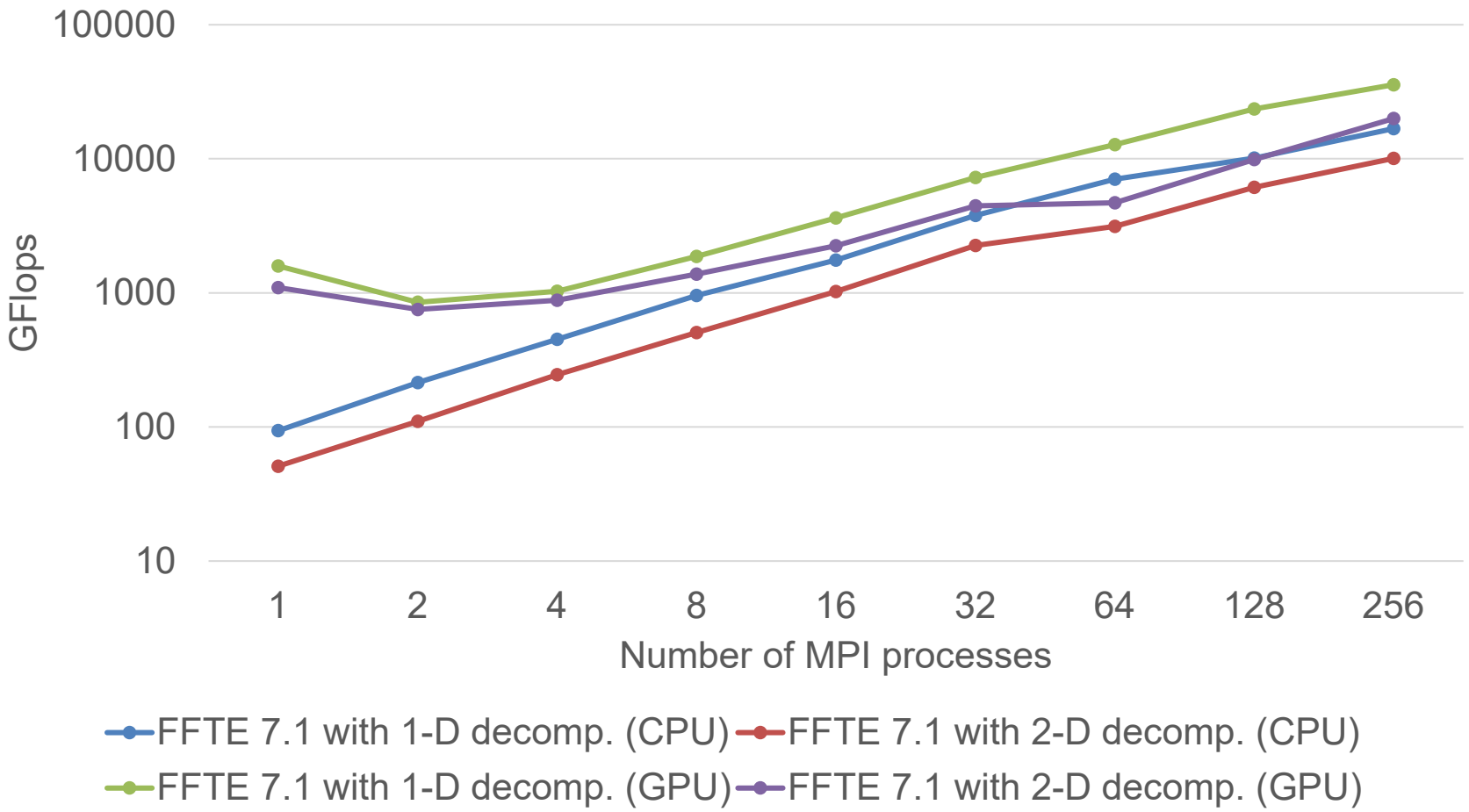
Performance Results

- To evaluate the parallel 3-D FFTs, we compared
  - FFTE (version 7.1, CPU) with 1-D decomposition
  - FFTE (version 7.1, CPU) with 2-D decomposition
  - FFTE (version 7.1, GPU) with 1-D decomposition
  - FFTE (version 7.1, GPU) with 2-D decomposition
- Weak scaling ($N = 512 \times 512 \times 512 \times$ MPI processes) and strong scaling ($N = 512 \times 512 \times 512$) were measured.

Evaluation Environment

- Miyabi-G at Joint Center for Advanced HPC (JCAHPC).
  - 1120 nodes, Peak 78.8 PFlops
  - CPU: NVIDIA Grace (72 cores, 3.0 GHz, 3.456 TFlops)
  - GPU: NVIDIA H100 (66.9 TFlops in FP64 Tensor Core)
  - Interconnect: InfiniBand NDR
  - Compiler: NVIDIA HPC Compilers 24.9
  - MPI library: OpenMPI 4.1.7a1
  - Compiler option: “-fast -mp” (for CPU)
    “-fast -cuda -gpu=cc90” (for GPU)
- Each MPI process has 72 cores and 72 threads, i.e. 1 MPI processes per node.
- The NVIDIA CUDA FFT library (CUFFT) is called to perform multicolumn FFTs on GPU implementation.

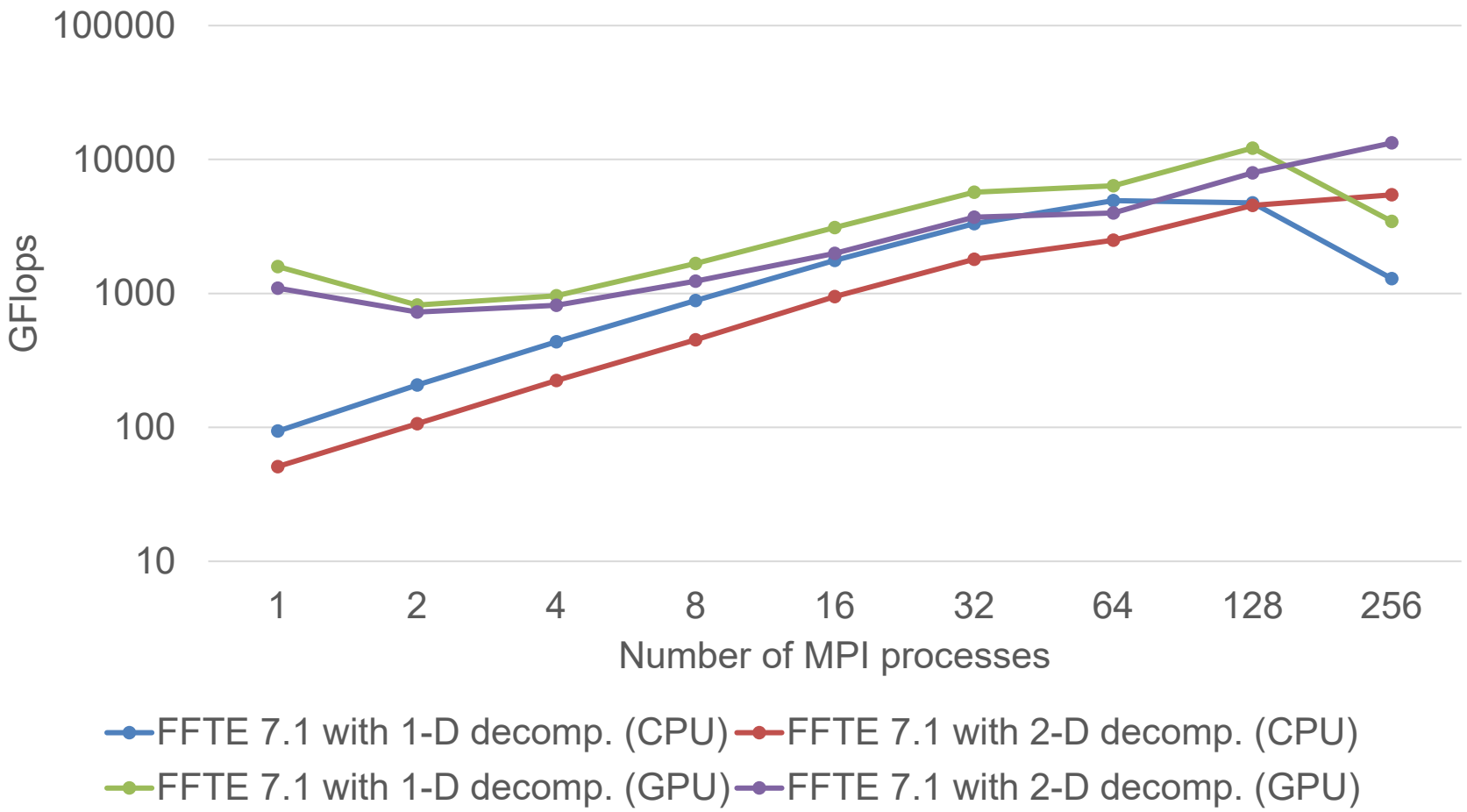
Performance of Parallel 3-D FFTs ($N = 512 \times 512 \times 512 \times$ MPI processes)



Discussion (1/2)

- In the case of weak scaling, the performance of one-dimensional decomposition is better than that of two-dimensional decomposition for both CPU and GPU implementations.
- This is because that the total communication amount of the one-dimensional decomposition is half that of the two-dimensional decomposition.

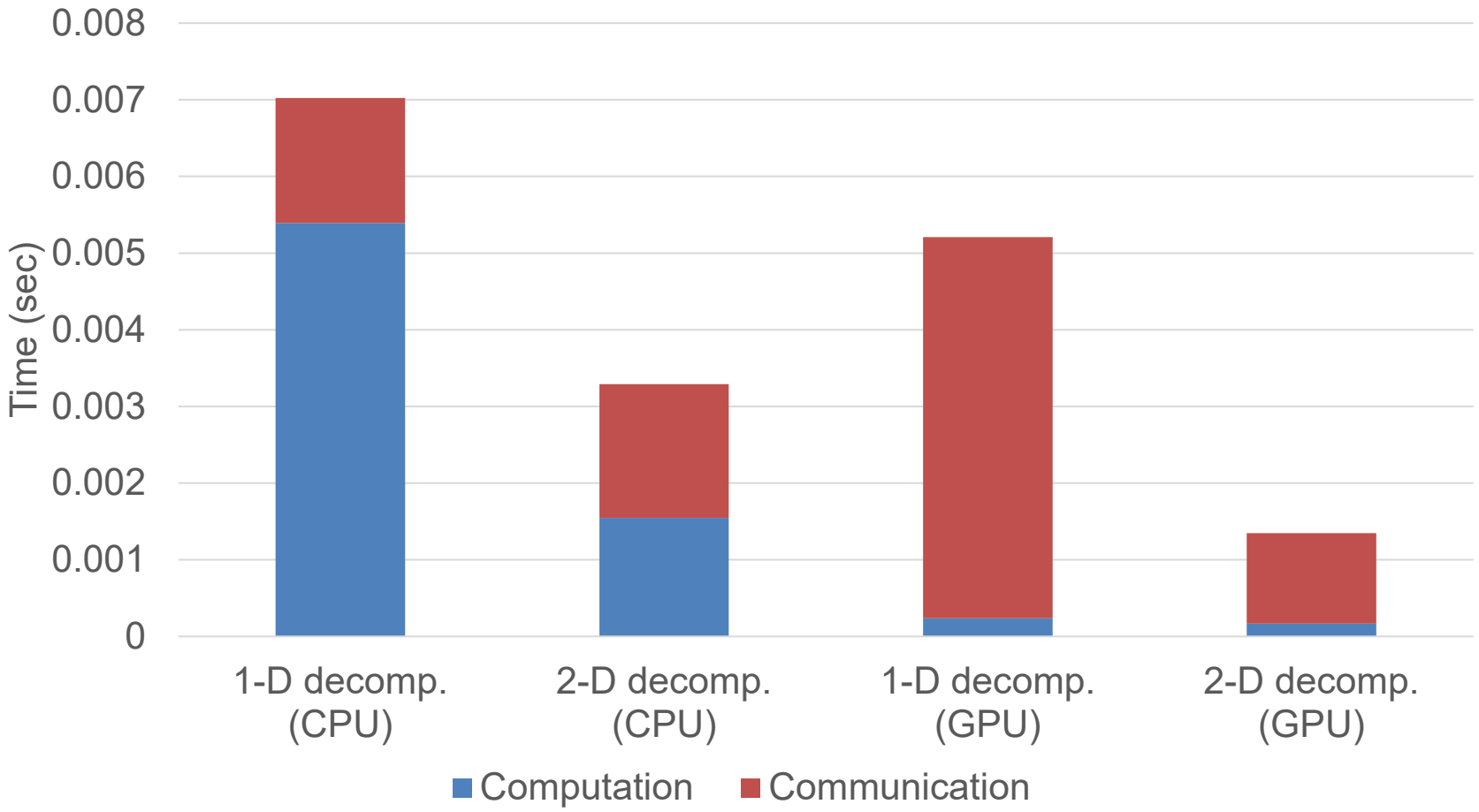
Performance of Parallel 3-D FFTs ($N = 512 \times 512 \times 512$)



Discussion (2/2)

- In the case of strong scaling, the performance of one-dimensional decomposition is better than that of two-dimensional decomposition for both CPU and GPU implementations when the number of MPI processes is 128 or less.
- On the other hand, for 256 MPI processes, two-dimensional decomposition is faster than one-dimensional decomposition for both CPU and GPU implementations.
- This is because for a 512^3-point FFT, the message size for all-to-all communication is only 32 KB with one-dimensional decomposition, whereas it becomes 1 MB and 2 MB with two-dimensional decomposition.

Breakdown of Execution Time in FFTE 7.1 ($N = 512^3$, 256 MPI processes)



Conclusion

- We proposed an implementation of parallel 3-D FFT with 2-D decomposition on GPU clusters.
- We showed that a 2-D decomposition effectively improves performance by reducing the communication time for larger numbers of MPI processes.
- The performance results demonstrate that the proposed implementation of a parallel 3-D FFT with 2-D decomposition is efficient for improving the performance on GPU clusters.