Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures
Final Report for JHPCN Joint Research of FY 2025

jh250041

Study on the real effect of non-blocking collective communications

Takeshi Nanri（Kyushu University）

Abstract

Non-Blocking Collective communications are expected to be a means to overlap collective communication with communication and hide the communication time. This project investigates the usage and the real effect of NBCs in JHPCN resources so that programmers can make correct decisions to use them or not. Based on the findings in the last year, we have examined the effect of overlapping with progress threads and confirmed its effective to hide large message communications. Also, we added a mechanism for adjusting computation amount to our benchmark NBC-Eff so that the users can compare the effect of overlapping among different implementations of NBCs easily. As for applications, we have applied thread-based overlapping of collectives and experienced sufficient effects.

1. Basic Information

(1) Collaborating JHPCN Centers

Hokkaido University

The University of Tokyo

Nagoya University

Kyoto University

Osaka University

Kyushu University

(2) Theme Area

Large scale computational science area

(3) Project Members and Their Roles

(4) Takeshi Nanri[1]:PI, Topic 1 and 2
Kengo Nakajima[2]: Co-PI, Topic 3
Richard Wilson Vuduc[3]: Co-PI, Topic 3
Takeshi Fukaya[4]: Topic 3
Osamu Tatebe[6]: Topic 3
Daisuke Takahashi[6]: Topic 3
Toshihiro Hanawa[2]: Topic 1, 2 and 3
Shinji Sumimoto[2]: Topic 1, 2 and 3
Takahiro Katagiri[8]: Topic 1, 2 and 3
Keiichiro Fukazawa[16]: Topic 1 and 3
Susumu Date[10]: Topic 1 and 2
Takashi Soga[10]: Topic 1 and 2
Yoshiyuki Morie[11]: Topic 1 and 2
Kaushik Kandadi suresh[14]: Topic 1 and 2
Benjamin Michalowicz[14]: Topic 1 and 2
Tu Tran[14]: Topic 1 and 2
Bharath Ramesh[14]: Topic 1 and 2
Gerhard Wellein[16]: Topic 3
Gerardo Cisneros-Stoianowski[12]: Topic 1 and 2
Brody Williams[12]: Topic 1 and 2
Takeo Narumi[1]: Topic 1 and 2
Shoji Sakoda[9]: Topic 1 and 2
Remma Arisako[1]: Topic 1 and 2
Souvadra Hati[3]: Topic 3
Jacob D. Tronge[3]: Topic3
Takeru Narumi[1]: Topic 2
Wataru Miyasaka[1]: Topic 1
1: Kyushu U., 2: U. Tokyo, 3: Georgia Inst. of Tech., 4: Hokkaido U., 6: U. Tsukuba, 7: Inst. of Science Tokyo, 8: Nagoya U., 9: Kyoto U., 10: Osaka U., 11: Teikyo U., 12: NVIDIA Co., 13, Technical U. of Munich, 14: Ohio State U., 15: Technische U. Darmstadt, 16: National Inst. for the Humanities

2. Purpose and Significance of the Research

NBCs (Non-Blocking Collective communications) are expected to be a means to overlap collective communication with computation and hide the communication time. This ability was adopted in the MPI-3.0 standard approved in 2012 and has since been implemented in most MPI libraries. However, the use of NBCs is currently limited because programmers do not have sufficient information about the usage and effect to modify their algorithms to overlap communication and communication.

Therefore, in this project, we are

investigating the usage and the real effect of NBCs in JHPCN resources. In addition to examining existing benchmark programs, we are applying NBCs on several scientific and technical computing algorithms and verifying whether communication hiding improves their performance. Note that NBCs is defined broadly in this study. In addition to collective communications de-fined in MPI standard, such as MPI_Iallreduce, non-blocking communica-tions with common pattern such as halo-exchange are included as our targets.

The significance of this research is to pro-vide programmers with correct knowledge about the usage and performance charac-teristics of NBCs and the effect of commu-nication hiding in real applications. This will enable programmers to make correct decisions to actively use NBCs when the effect of communication hiding is expected, and to avoid using it when the effect cannot be expected, depending on their own algo-rithms and the HPC systems they use.

3. Significance as JHPCN Joint Research Project

Achieving the communication hiding effect of NBCs requires knowledge from computer scientists about the configuration of the computer used, available communication libraries and their appropriate use, and knowledge from computational scientists about the algorithms that overlap computation and communication. For this reason, the JHPCN is well suited as such an interdisciplinary research effort. In addition, since a wide variety of parallel computers are available at JHPCN, it is expected that the knowledge gained from this project will be useful for many (if not all) HPC systems worldwide.

4. Outline of Research Achievements until FY2024

In JH240058, we studied the current state of NBCs as well as their basic performance. In Topic 1, we checked available communication hiding techniques on each JHPCN resource. Also, OSU members modified their MVAPICH2 code to enable SHARP with Fujitsu's job scheduler, and NVIDIA members modified their HPC-X code to enable two or more PPNs with SHARP. In Topic 2, we measured the overlap ratio with NBCs on each JHPCN resource with OSU Micro-Benchmarks. Also we have built a benchmark program NBC-Eff so that we can compare the effects of communication hiding techniques. In Topic 3, we have investigated the effects of mixed/low precision computing on pipelined conjugate gradient (CG) methods. Pipelined CG (PiCG) is known to be more susceptible to rounding errors than original CG, and it is unstable for procedures with mixed/low precision computing [Nakajima et al. 2018]. We have evaluated the effects of Residual Replacement [van der Horst and Ye, 2000] and Iterative Refinement [Strzodka et al. 2006] for stabilization of PiCG with mixed/low precision computing. In addition to that, we explored other communication hiding algorithms that should be investigated in FY2025.

5. Details of FY2025 Research Achievements

Topic 1: Study on the effective usage of NBCs

In FY2024, we have found that there are limitations of PPNs (the number of processes per node) in progress methods, especially the offloading mechanism SHARP, for NBCs. Therefore, in this year, we examined two new approaches, 1) hiearchical NBC algorithm, and 2) multi progress-threads per core technique.

As for 1), we designed prototypes of hierarchical algorithms for MPI_Ibcast and MPI_Iallreduce. They divide the communication into intra-node layer and inter-node layer so that only one process per node participate inter-node communication. This algorithm is implemented by dividing the MPI_COMM_WORLD communicator into subcommunicators according to the rank allocation. To examine the effect of this implementation, its performance is examined with and without enabling SHARP. We have expected that, by limiting only one process per node to participate inter-node collective communication, our approach can achieve effects of SHARP even with multiple PPNs. However, the result did not show significant effect of SHARP. In our investigation for its reason, we found that SHARP is only applied for the MPI_COMM_WORLD communicator and does not work on the subcommunicators.

As for 2), we binded multiple progress threads in one CPU core to enable calculation threads to use as much cores as possible. This technique is enabled by applying pthread_setaffinity_np function after creating a progress thread. The core to bind is chosen according to the NUMA structure. As a preliminary evaluation, we measured the effect of this technique on flat-MPI execution. So, we compared the

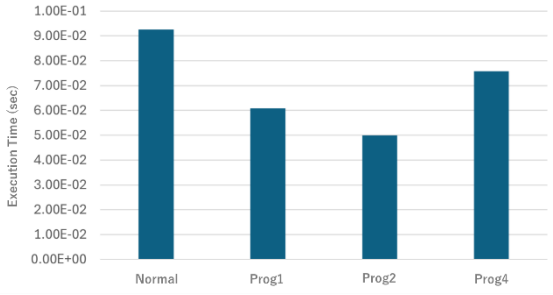

Figure 1 Effect of binding progress threads to one core (16B Allgather + computation ratio=1/2)

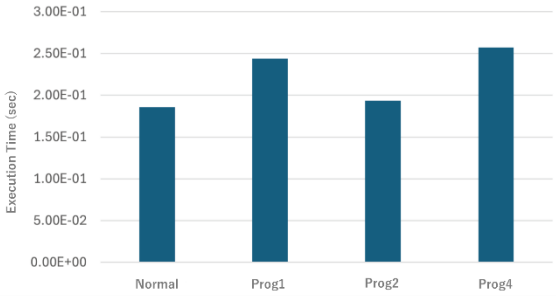

Figure 2 Effect of binding progress threads to one core (16B Allgather + computation ratio=2)

execution time among the following four cases.

Normal: Run without overlap

Prog1: Run with a progress thread per process with one thread per core

Prog2: Run with a progress thread per process with binding two threads to one core

Prog4: Run with a progress thread per process with binding four threads to one core

Experiments are performed by overlapping MPI_Allgather and matrix-multiplication on Genkai node-group A with 32 nodes. The numbers of calling matrix-multiplication is adjusted so that the ratios of computation time will be 1/2 for Small-comp case, and 2 for Large-comp case. The number of processes per node are decided according to the available number of cores for computation. So, Normal runs 120

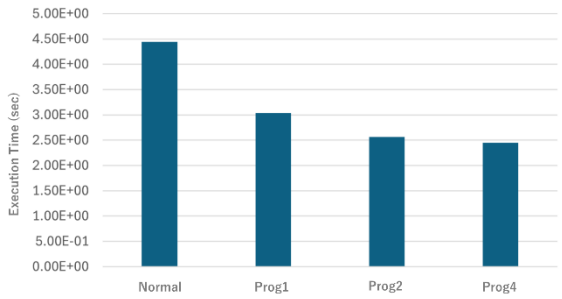

Figure 3 Effect of binding progress threads to one core (1MB Allgather + computation ratio=1/2)

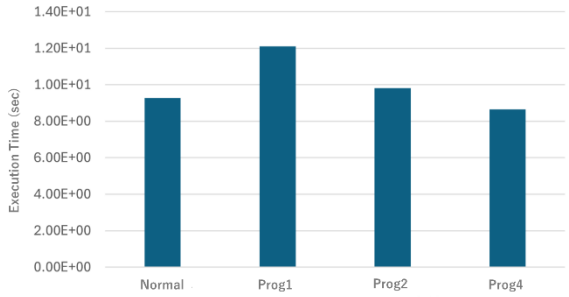

Figure 4 Effect of binding progress threads to one core (1MB Allgather + computation ratio=2)

processes per node, while Prog1, Prog2 and Prog4 run 60, 80 and 96 processes per node respectively. Figure 1 and 2 show the results of message size 16 byte for Small-comp case and Large-comp case, respectively. Figure 1 shows that Prog2 outperforms the others. This means that the effect of overlapping is achieved by binding two progress threads to one core. On the other hand, Prog4 is not as effective as Prog2 because of the overhead for context switch among four progress threads on one core. In Figure 2, as the computation ratio over communication increases, Normal becomes the fastest because it can use all 120 cores per node for computation, while Prog1, Prog2 and Prog4 can use less number of cores for computation because they consume some cores for communication. Figure 3 and 4 show the results of message size 1MB. Figure 3 shows that Prog4 outperforms the others. So, as the message size increases, the overhead of context switch becomes negligible. Also, as shown in Figure 4, Prog4 is faster than Normal. Therefore, in this case, even when the computation time is longer than the communication time, progress thread can achieve overlapping effect by binding four progress threads to one core.

In this year, we also started to examine the effect of NBCs on GPU clusters. Normally, since GPUs cannot directly invoke MPI functions, GPU applications needed synchronization between CPU and GPU, such as cudaDeviceSynchronize, before MPI calls on CPU. We built an alternative method in which GPU kernel sends a notification to a CPU thread to start communication so that synchronization can be avoided. In addition to that, to reduce the number of kernel invocation, we constructed a CUDA Graphs by combining a compute kernel and a notification kernel.

Figure 5 shows the result of an experiment that performs parallized 4-point NxN 2D stencil calculation on 4 GPUs of GENKAI node-group B. Computations are divided by row with blocks. The following four methodsare compared:

Default: Synchronize CPU and GPU
Notify: Use Notify kernel
Coop: Use cooperative groups
Graphs: Proposed method
(CUDA Graphs + Notify)

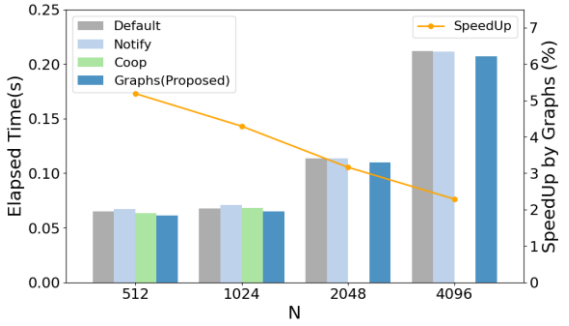

Figure 5 Comparison among GPU communication methods with NxN stencil

For all matrix sizes, the proposed method performed better than other ones. This is because our method could reduced the overhead for confirming completion of write to the device memory.

Topic 2: Study trends of communication hiding by NBCs

In FY2024, we have built a benchmark program, NBC-Eff, that can compare the effect of overlapping on the performance of applications among different NBC mechanisms. This year, to ease the users, we have added a mechanism that adjusts computation amount to overlap automatically according to the communication speed of all target NBC mechanisms. It measures the communication time for each candicate implementation and calculates the range of the computation amount sufficient enough to examine the overlap effects of them. Then, it divides the range into 10 and measures the execution time for each implementation. Figure 6 shows the results of the benchmark for Intel MPI and HPC-X with 1 process per node. In this case, HPC-X with SHARP enabled outperforms other ones. On the other hand, Figure 7 shows the similar

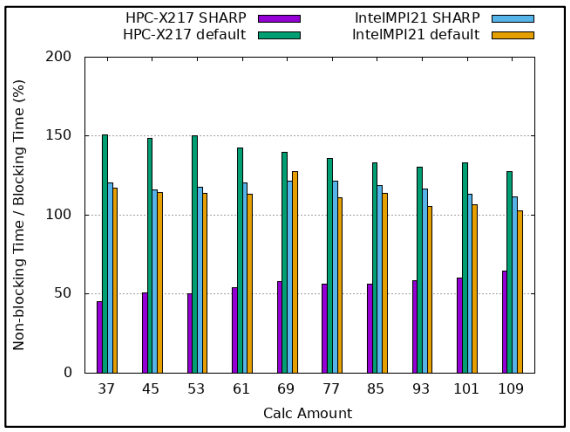

Figure 6 Comparison of the overlap effect of MPI_Iallreduce between Intel MPI and HPC-X (1 proc / node)

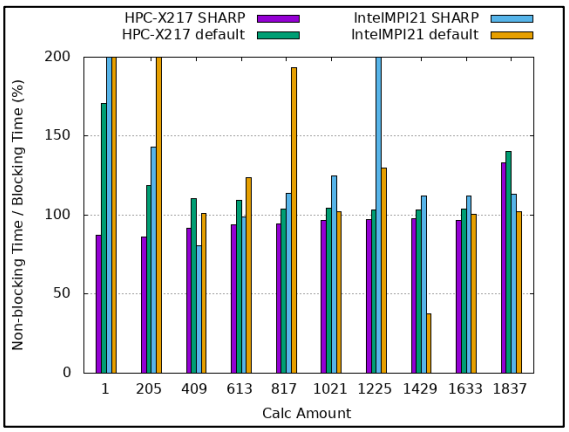

Figure 7 Comparison of the overlap effect of MPI_Iallreduce between Intel MPI and HPC-X (8 procs / node)

comparison with 8 processes per node. In this case, SHARP does not show significant advantage. Also, as the calculation amount increases, Intel MPI shows better performance than HPC-X. These results show that our benchmark can be used to fairly compare the real effect of overlapping by NBCs.

Topic 3: Investigation of communication hiding algorithms with non-blocking collective communication

In computation of sparse tensor decompositions, we are looking at a pattern of communication required for MTTKRP

(Matricized Tensor Times Khatri Rao Product) operations within an existing implementation called SPLATT. Since this pattern is not directly supported by MPI collectives we are examining some implementations for it. We would like to develop a better understanding of how sparse tensor decompositions perform in various environments and hardware configurations.

For CG method, we primarily investigated pipeline-type algorithms. In [中島 et al., May 2025], we examined single-precision and mixed-precision approaches and demonstrated that mixed precision with Residual Replacement (RR) achieves convergence with nearly the same number of inner iterations as double-precision computations, if the problem is well-conditioned. When applying single precision with Iterative Refinement (IR), performance was favorable for well-conditioned problems (approximately twice the efficiency per iteration); however, for ill-conditioned problems, it was inferior to mixed precision with RR (which achieved about 1.33× efficiency). The combination of single precision with IR and RR outperformed single precision with IR for well-conditioned problems but degraded performance for ill-conditioned cases.

In [中島 et al., Aug. 2025], we addressed the pipelined CG method, which hides communication in inner products, and proposed an implementation strategy that is independent of specific hardware, compilers, or MPI libraries in large-scale parallel environments. Our approach assigns collective communication for inner products and halo exchanges to the master thread and overlaps them with sparse matrix-vector multiplications on pure interior points. Using up to 2,048 nodes of Wisteria/BDEC-01 (Odyssey), this method achieved a 27% performance improvement compared to no CC-overlapping and about a 6% improvement compared to applying CC-overlapping only to halo communication.

For FFT, we investigated the overlap between computation and communication in the all-to-all communication, which is the bottleneck in parallel FFT, on a GPU cluster. However, we found that the communication time significantly exceeds the computation time, limiting the effectiveness of overlap.

For global atmospheric simulation code, we investigated the parts where MPI_Allgatherv is used and examined the computations that can be overlapped with the communication. As a result, we identified the pressure calculation, advection calculation, transformation to the wavenumber space, and I/O operations as sections where overlap is possible. Except for the I/O operations, these parts are executed in every iteration loop, and thus are expected to be effective targets for overlap optimization. The I/O section, although highly dependent on the problem size, is currently implemented such that rank 0 performs collective output. Under this structure, a certain degree of benefit from overlap can also be expected.

6. Self-review of Current Progress and Future Prospects

Based on the results in FY2024, we examined other approaches for overlapping

collective communications. Through the experiments of hierarchical algorithms of collective communications, we learned that current MPI library does not use SHARP for subcommunicators. Also, the results of binding multiple progress threads to one core showed its possibility of overlapping as well as its bottleneck for context switch. By combining CUDA Graphs and our notification kernel, we studied the effect of GPU initiated communication. By adding a mechanism for auto-adjustment of computation amount, our benchmark has become more convenient to compare the overlap effect among different NBC implementations.

Overall, we can conclude that we have achieved sufficient results of study on the effect of NBCs.

In FY2026, we are tackling with the following research items:

- Develop a thread compaction mechanism to enable more efficient progress threads
- Applying CUDA Graphs + Notify kernel to overlap collective communications in GPU applications
- Performance analysis of NBC implementations using NBC-Eff benchmark
-

#Section 7, achievements of this FY, should be input on the JHPCN website.