jh240058

Study on the real effect of non-blocking collective communications

Takeshi Nanri（Kyushu University）

Abstract

In this project, we have investigated availability of communication overlapping mechanisms for NBC on JHPCN computer resources (Topic 1), studied the effects of overlapping with each mechanism (Topic 2), and examined some applications as target of overlapping with NBC (Topic 3) in FY2024. From the results of Topic 1 and 2, we found that the current implementations of NBC provide only limited benefits for communication overlap. As the number of processes per node increases, the overlap ratio has fallen down significantly with NVIDIA SHARP. Though Progress Thread showed high overlap ratio with larger number of processes per node, the fundamental communication speed is much slower than the cases with Progress Thread disabled. In Topic 3, some applications have shown possibility of taking advantage of overlapping with NBC. Considering these results, in addition to the original plan, we consider improving the implementation techniques of NBC in FY2025.

1. Basic Information

(1) Collaborating JHPCN Centers

Hokkaido University

Tohoku University

The University of Tokyo

Institute of Science Tokyo

Nagoya University

Kyoto University

The University of Osaka

Kyushu University

(2) Theme Area

Large scale computational science area

(3) Project Members and Their Roles

Takeshi Nanri[1]:PI, Topic 1 and 2
Kengo Nakajima[2]: Co-PI, Topic 3
Takeshi Fukaya[4]: Topic 3
Hiroyuki Takizawa[5]: Topic 1 and 2
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
Maddegedara Lalith[2]: Topic 3
Rio Yokota[7]: Topic 1 and 3
Takahiro Katagiri[8]: Topic 1, 2 and 3
Keiichiro Fukazawa[9]: 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
Bengisu Elis[13]: Topic 1 and 2
Dennis Herr[13]: 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
Fabian Czappa[15]: Topic 3
Ayesha Afzal[16]: Topic 3
Takeo Narumi[1]: Topic 1 and 2
Kazuya Yamasaki[2]: Topic 3
Shiyao Xie[7]: Topic 2

1: Kyushu U., 2: U. Tokyo, 4: Hokkaido U., 5: Tohoku U., 6: U. Tsukuba, 7: Inst. of Science Tokyo, 8: Nagoya U., 9: Kyoto U., 10: U. Osaka, 11: Teikyo U., 12: NVIDIA Co., 13, Technical U. of Munich, 14: Ohio State U., 15: Technische U. Darmstadt, 16: NHR

2. Purpose and Significance of the Research

NBC (Non-Blocking Collective communication) is expected to enable overlapping collective communication with computation and hide the communication time. However, its usage is currently limited to a small number of applications. It is mainly because of the insufficient information such as its usage and effect. Since overlapping communication and computation requires algorithm modification, programmers need to know how to use it and make sure if it can speed up their applications.

To provide such information, this project addresses the following three topics:

Topic 1: Investigate the availability and usage of NBC mechanisms on each computing resource of JHPCN.

Topic 2: Measure the effect of communication hiding by NBC mechanisms on each computer resource using benchmark pro-grams. Also, study their performance characteristics.

Topic 3: Examine the effect of communication overlapping on the performance of applications. In addition to the existing algorithms, we will develop new algorithms of communication overlapping for our ap-plications.

The significance of this research is to provide programmers with correct knowledge about the usage and the effect of communication hiding with NBC so that they can decide if they should try it in their applications or not.

3. Significance as JHPCN Joint Research Project

Achieving the communication hiding effect of non-blocking collective communication 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 FY2023

This project is not a continuous project.

5. Details of FY2024 Research Achievements

Topic 1:

In Topic 1, members from each JHPCN computer center, NVIDIA Co. and Ohio State University worked together to investigate available communication overlapping mechanisms on each JHPCN resource. Table 1 shows the results.

Table 1 Available NBC mechanisms

System	Interconnect	Available NBC methods	Overlap effect
AOBA-S	InfiniBand NDR	SHARP	SHARP
Wisteria-O	Tofu-D	Tofu Barrier, Assistant Core	Assistant Core
TSUBAME4.0	InfiniBand NDR	SHARP, Progress Thread	SHARP, Progress Thread
Flow Type I	Tofu-D	Tofu Barrier, Assistant Core	Assistant Core
Flow Type II	InfiniBand EDR+HDR	Progress Thread	Progress Thread
Camphor3	InfiniBand NDR + HDR (storage)	SHARP, Progress Thread	In study
SQUID	InfiniBand HDR	SHARP, Progress Thread	In study
Genkai	InfiniBand NDR	SHARP, Progress Thread	SHARP, Progress Thread

Followings are the summary of the results:

- With careful adjustments of middleware and MPI libraries, NVIDIA SHARP showed the overlap effect on some InfiniBand-based systems. Systems with "In study" as the overlap effect in the table requires more time for such adjustments. Also, we have investigated that the overlap effect by SHARP degrades rapidly as the number of processes per node becomes four or more (see Topic 2 for detail).
- Progress Threads mechanism showed high overlap ratio with MVAPICH2 and MVAPICH 3.0 even when the number of processes per node is four or more. However, enabling this mechanism reduces the fundamental communication speed significantly (see Topic 2 for detail).
- Assistant core enables high overlap ratio with four or more processes per node. To achieve this effect, programmers need to specify the range for using this mechanism in the program. On the other hand, Tofu Barrier is not currently used in the implementation of NBC functions of MPI library.

As a result of the investigation in this topic, we found that in the current implementation of the MPI libraries, the effect of communication hiding using NBC functions is very limited. Therefore, we have decided to include a study consider improving implementation techniques of NBC functions in the research plan for FY2025.

Topic 2:

We have studied the effects of different overlapping mechanisms of NBCs.

As a first step, we compared the overlap ratio of different NBC implementations by using two open-source benchmark suites, OMB (OSU cro Benchmarks) and IMB (Intel MPI Benchmarks). Figure 1 shows the result of osu_iallreduce program in OMB with 128 nodes of Genkai. In this figure, SHARP shows more than 80% overlap ratio with HPC-X 2.17.1 when 1 proc/node, while the ratio goes down when the number of processes per node becomes 4. On the other hand, Progress Thread shows more than 90% overlap ratio with Intel MPI and MVAPICH even with 4 procs/node.

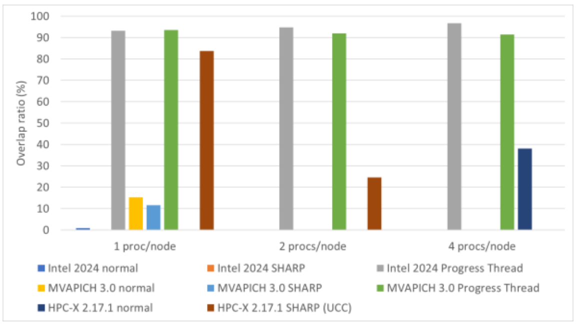

Figure 1 Overlap ratio on Genkai (osu_iallreduce, 32 bytes, 128 nodes)

Also, NBC benchmark of IMB was used to measure the communication overlapping effect on TSUBEME 4.0 from 1 to 16 nodes (1 MPI process per node) with and without SHARP, respectively. We measured the performance of the collective communications (MPI_Ireduce, MPI_Iallreduce, and MPI_Ibarrier) supported by SHARP. When 8 MPI and 16 MPI processes were used, the overlap rate was less than 30% without SHARP when the message size was 16KB or larger in MPI_Ireduce. In contrast, when SHARP was used, the overlap rate was more than 80%, indicating that SHARP is effective for communication concealment.

Since benchmark programs for NBCs in OMB and IMB are designed to measure the maximum overlap ratio, they automatically adjust the amount of computation to be sufficient to overlap communication. This makes it difficult to study the effect of NBC on the total performance of applications with them.

Therefore, as a method to investigate the effect of NBC on the application performance, we are designing a new benchmark program NBC-Eff. It fixes the amount of computation to overlap communications so that we can check how the overlapping could reduce (or increase) the total execution time by applying NBCs with different methods.

Figure 2 shows a result of its prototype. By comparing the execution times of "No-overlap" and "Overlap" we could see that, by default overlapping cannot reduce the execution time, while it can speed up execution by using overlapping mechanisms such as Progress Thread and SHARP.

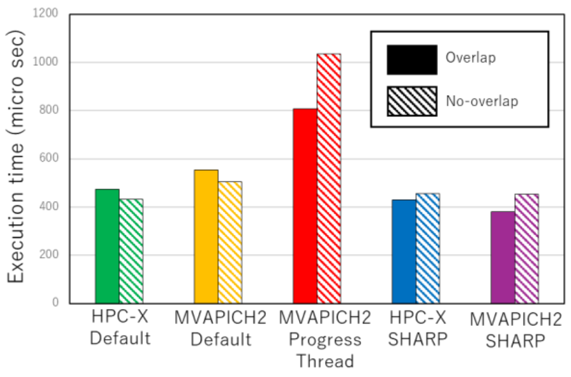

Figure 2 Result of NBC-Eff (8 byte MPI_Iallreduce, 32 nodes x 8 procs)

On the other hand, by comparing the methods for NBCs, the execution time with Progress Thread was much longer than other cases even with overlapping. It is mainly because enabling Progress Thread requires MPI library to run in MPI_THREAD_MULTIPLE mode for thread-safety, which introduces extra overhead in most of MPI function. This result confirms that we cannot rely on the results of overlap ratio to investigate the effect of overlapping.

To analyze the side-effects of overlapping, we have also developed a performance model for non-blocking collective communication that considers the impact of memory performance limits. This assumes that when overlapping communication and computation in high throughput, their performance is limited by memory bandwidth. Therefore, we conducted basic experiments to model the performance of CPUs/GPUs and communication devices using memory bandwidth as a basic unit of resources. In this experiment, we increased the number of threads performing calculations to increase memory access and we confirmed the overlap ratio with communication. Figure 3 shows the overlap ratio when the number of threads is increased during overlapped execution.

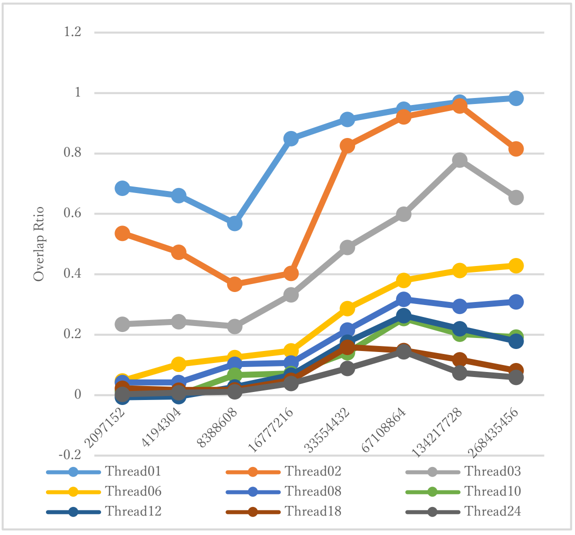

Figure 3 Effect of memory bandwidth on the overlap ratio

As the graph shows, the overlap ratio decreased as the number of threads increased. This confirmed that the upper limit of memory bandwidth is affecting the effect of NBC.

Topic 3:

In Topic 3, several applications are examined as targets of communication overlapping.

```
Compute $r^{(0)}=b-[A]x^{(0)}$;solve$[M]u^{(0)}=r^{(0)}$;
$p^{(0)}=u^{(0)}$;$s^{(0)}=[A]p^{(0)}$; $\gamma_0= r^{(0)}u^{(0)}$
for i= 1, 2, …
    $\delta_{i-1}= p^{(i-1)}s^{(i-1)}$
    solve $[M]q^{(i-1)}= s^{(i-1)}$
    $\alpha_{i-1}= \gamma_{i-1}/ \delta_{i-1}$
    $x^{(i)}= x^{(i-1)} + \alpha_i p^{(i)}$
    $r^{(i)}= r^{(i-1)} - \alpha_i s^{(i-1)}$
    $u^{(i)}= u^{(i-1)} - \alpha_i q^{(i-1)}$
    check convergence |r|
    $\gamma_i = r^{(i)}u^{(i)}$
    $w^{(i)}= [A]u^{(i)}$
    $\beta_i = \gamma_i/\gamma_{i-1}$
    $p^{(i)}= u^{(i)} + \beta_i p^{(i-1)}$
    $s^{(i)}= w^{(i)} + \beta_i s^{(i-1)}$
end
```

(a) Original Pipelined CG

```
Compute $r^{(0)}=b-[A]x^{(0)}$;solve$[M]u^{(0)}=r^{(0)}$;
$p^{(0)}=u^{(0)}$;$s^{(0)}=[A]p^{(0)}$; $\gamma_0= r^{(0)}u^{(0)}$; y=0
for i= 1, 2, …
    $\delta_{i-1}= p^{(i-1)}s^{(i-1)}$
    solve $[M]q^{(i-1)}= s^{(i-1)}$
    $\alpha_{i-1}= \gamma_{i-1}/ \delta_{i-1}$
    $x^{(i)}= x^{(i-1)} + \alpha_i p^{(i)}$
    $r^{(i)}= r^{(i-1)} - \alpha_i s^{(i-1)}$
    $u^{(i)}= u^{(i-1)} - \alpha_i q^{(i-1)}$
    if time to replace
        $y= y + x^{(i)}$
        $r^{(i)}= b - Ay$
        $x^{(i)}=0$
        solve $[M]u^{(i)}= r^{(i)}$
    endif
    check convergence |r|
    $\gamma_i = r^{(i)}u^{(i)}$
    $w^{(i)}= [A]u^{(i)}$
    $\beta_i = \gamma_i/\gamma_{i-1}$
    $p^{(i)}= u^{(i)} + \beta_i p^{(i-1)}$
    $s^{(i)}= w^{(i)} + \beta_i s^{(i-1)}$
end
$x= x^{(end)} + y$
```

(b) Pipelined CG with RR

Figure 4 Residual-Refinement (RR) for Pipeline Conjugate Algorithm (Additional RR part in red letters is computed by double precision arithmetic)

- Pipelined CG

In large-scale parallel computing environments, Krylov subspace methods such as the conjugate gradient method often experience significant performance degradation due to parallel overhead. The primary factors contributing to this issue include matrix-vector multiplications (point-to-point communication) and inner products (collective communication). This study focuses on addressing the latter challenge by investigating the pipeline conjugate gradient method. While the pipeline conjugate gradient method follows the same algorithmic structure as the conventional conjugate gradient method, modifications to the computation order alter the propagation of rounding errors. It is well-known that when low-precision or mixed-precision arithmetic is introduced, the method may suffer from stagnation or deterioration in convergence. In the present work, we explore stabilization techniques, including Residual Replacement (RR) and Iterative Refinement (IR), for solving systems of linear equations arising from three-dimensional heat conduction equations using the finite element and finite volume methods (Figure 4).

These approaches aim to enhance the robustness of mixed-precision pipeline conjugate gradient methods and mitigate the adverse effects on convergence. Iterative Refinement method combined with Residual Replacement (IR-RR) provides pipeline conjugate gradient method with low-precision robust convergence. We evaluated these effects using benchmark programs parallelized by OpenMP on a single node of Fujitsu A64FX architecture. In FY.2025, we extended idea for the applications on massively parallel distributed environments.

- Jacobi method

We have applied a communication overlapping method for neighbor-exchange data transfer in Jacobi method, using threads operation, and confirmed that it enables better performance than using MPI_Isend/Irecv (Figure 5)

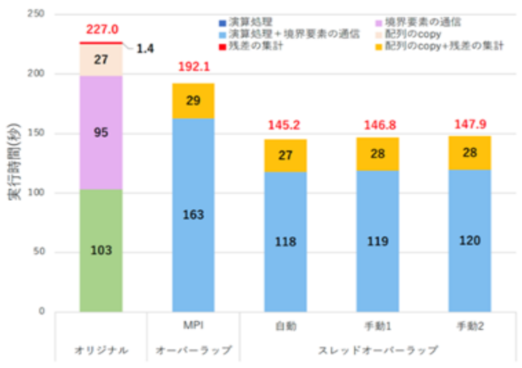

Figure 5 Comparison of overlapping methods between MPI_Isend/Irecv and thread

- MHD (Magneto-Hydro-Dynamics)

It is common in parallelized simulations, but in MHD simulation codes, there are cases where distributed data must be merged before writing out the calculation results (in this case, the MPI_file_write function is not utilized). To consolidate this distributed data, a gather operation is performed. The code was modified to overlap the gather operation with the file write operation, and the effects of this modification were investigated. The computational domain is a three-dimensional space of 600×400×400, and parallelization is achieved using Strong Scaling with one-dimensional domain decomposition in the z direction (utilizing 32, 48, and 64 processes). The measurement environment is Camphor3, and MPI is implemented using MVAPICH version 3.0.

To enable overlap, the MPI_Igatherv (asynchronous MPI_Gatherv) function was utilized, following a preliminary verification of its performance through the variation of computation size and the number of parallel processes, with subsequent measurement of communication time.

As shown in Figure 6, the communication performance exhibited an enhancement when the computation size was substantial (i.e., the communication volume was high) and the number of processes was minimal. This result is consistent with that of conventional synchronous collective communication.

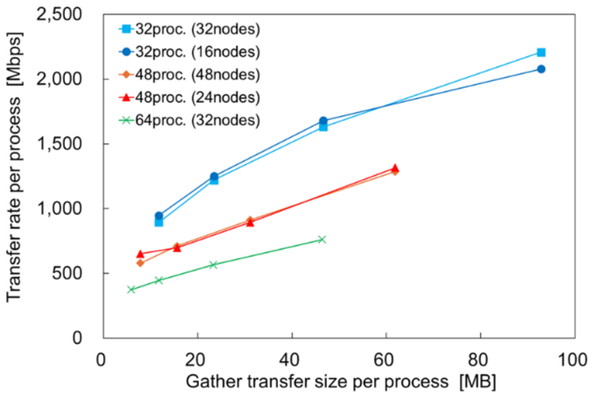

Figure 6 Performance evaluation of MPI_Igather transfer rate

Next, a comparison was performed between the conventional synchronous gather+write method and the overlapping method. As Figure 7 shows, there is a discrepancy in time between the synchronous version and the overlapping case. The positive difference suggests that overlapping is conducive to enhanced performance. Firstly, as the number of concurrent processes increases (i.e., from 32 processes), the overlapping method tends to provide better performance. Conversely, while overlap was anticipated to be efficient when the communication size was substantial and the communication time was extensive, no effect of overlap was detected. Conversely, the overlap effect was observed when the communication size was minimal. The precise reason for this phenomenon remains uncertain; however, it may be attributable to the equilibrium between the speed of data writing and the capacity for effective communication, or operational issues associated with asynchronous communication (MPI_Igather is called and subsequently wait finalization).

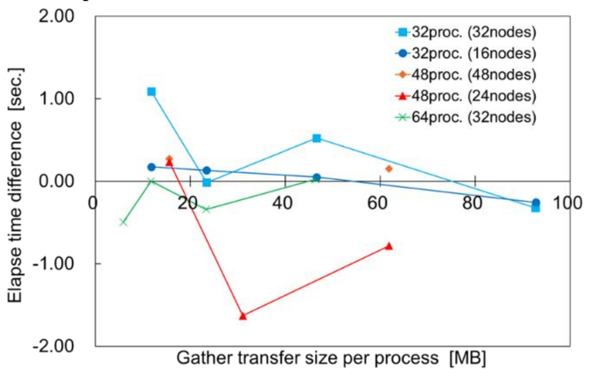

Figure 7 Performance evaluation of overlapping effect with MPI_Igather

- FFT (Fast Fourier Transform)

The communication hiding effect of parallel three-dimensional FFT with non-blocking collective communication (MPI_Ialltoall) is confirmed. Since MPI_Ialltoall is a bottleneck in parallel FFT, overlapping computation and communication can reduce execution time. We ran 8 MPI processes per node and 15 threads per node on 32 nodes of Genkai. When the Progress Thread was used, the number of computation threads was reduced to 14 threads. The results show that when the array size is large, the performance is about 10% higher when Progress Thread is enabled.

- WaitIO

In this year, before considering the overlapping, we conducted an initial evaluation of communication performance with blocking collectives in WaitIO. Specifically, the use of heterogeneous systems to integrate diverse workloads has become increasingly important in recent years. To enable efficient communication between heterogeneous systems, Information Technology Center, the University of Tokyo has developed h3-Open-SYS/WaitIO, a system that supports communication across different platforms.

WaitIO provides an MPI-like interface to facilitate migration from existing MPI programs, along with an implementation of collective communication using WaitIO. However, the current implementation of collective communication in WaitIO is not designed with heterogeneity in mind, and in systems with hierarchical communication performance, low-performance systems can become bottlenecks.

In this study, we focused on MPI_Allreduce, one of the most frequently used collective operations. We proposed and evaluated a hierarchical MPI_Allreduce algorithm that combines WaitIO with MPI. Performance evaluation on the Supercomputer "Flow" in Information Technology Center, Nagoya University, showed that the proposed method achieved up to 27.7 times speedup compared to the standalone MPI_Allreduce implementation using WaitIO.

6. Self-review of Current Progress and Future Prospects

The investigations planned for the first year of this project have been mostly completed. However, the finding that the current implementations of NBC provide only limited benefits for communication overlap was an unexpected surprise. Based on this result, in FY2025, in addition to the originally planned development of benchmark programs, implementation of overlapping in applications, and investigation of communication overlapping techniques in GPU applications, we have decided to consider improving the implementation techniques of NBC.