Post-print of: Proficz J., Sumionka P., Skomiał J., Semeniuk M., Niedzielewski K., Walczak M. (2020) Investigation into MPI All-Reduce Performance in a Distributed Cluster with Consideration of Imbalanced Process Arrival Patterns. In: Barolli L., Amato F., Moscato F., Enokido T., Takizawa M. (eds) Advanced Information Networking and Applications. AINA 2020. Advances in Intelligent Systems and Computing, vol 1151. Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-44041-1_72

# Investigation into MPI all-reduce performance in a distributed cluster with consideration of imbalanced process arrival patterns*

No Author Given

**Abstract** The paper presents an evaluation of all-reduce collective MPI algorithms for an environment based on a geographically-distributed compute cluster. The testbed was split into two sites: CI TASK in Gdansk University of Technology and ICM in Warsaw University, located about 300 km from each other, both connected by a fast optical fiber Ethernet-based 100 Gbps network (900 km part of the PIONIER backbone). Each site hosted a set of 10 compute nodes interconnected locally by the InfiniBand switches with the traffic forwarded by specialized hardware: IBEX G40 - QDR InfiniBand RDMA based Extension Platform. A set of six all-reduce algorithms, consisting of two ring-based (including a PAP-aware pre-reduced ring), two binomial-tree based and two hierarchical ones, was tested for balanced and imbalanced process arrival patterns (PAPs). The results showed high and stable bandwidth with large data transmission latency of the branch connecting the remote sites (about 13 ms in comparison to 10 µs locally), and for the tested algorithms there was an advantage of hierarchical approach, and then binomial tree. Finally, we also observed some performance increase in PAP-aware solution in comparison to its regular counterpart. The main conclusion is that for the distributed cluster environment with imbalanced PAPs, there is a need for designing new hierarchical algorithms with PAP-aware support.

**Key words:** All-reduce, Process arrival pattern, Distributed cluster, MPI, HPC

* The authors would like to thank our centers' directors: prof. Henryk Krawczyk (CI TASK) and dr. Marek Michalewicz (ICM) as well as our admin teams, especially Bartosz Pliszka (CI TASK), for their continuous support for our work. We also would like to thank the companies: Vcinity, Inc. and 2CRSI SA for providing the InfiniBand extender hardware used in the the research.

## 1 Introduction

Distribution of computation is a leitmotif in HPC since the first supercomputer was connected to the global network. The typical approach covers creation of grid structures covering a federation of independent clusters. However, modern solutions enable connection of individual nodes on the long geographical distances, giving the glimpse of a local cluster built over the pan-continental infrastructure [8].

On the other hand, using such communication infrastructure, even with the latest optical fibers and switches, causes some serious constraints into the final solution. The currently used network protocols can introduce substantial overhead and not all applications can be ported for such environment. In this article we propose to take care of one of the most important collective operations, typically used within Message Passing Interface [2] (MPI) implementation: all-reduce.

The main contribution of this paper is an analysis of MPI all-reduce performance for InfiniBand over long distance Ethernet with and without imbalanced process arrival patterns (PAPs) for regular, pap-aware and hierarchical algorithms. The performed experiments showed the advantage of the hierarchical over the PAP-aware approach, with surprisingly good results of the butterfly algorithm.

The structure of the paper is as follows: the next section describes the background of the problem including the all-reduce operation, PAPs and the tested algorithms, section 3 presents the related works, section 4 provides the details about the testbed environment set up between two computer centers in Poland: CI TASK and ICM, section 5 describes the performed tests including the experimental results, and the last section presents the final conclusions as well as the planned future works.

## 2 Background

All-reduce is a one of the most used MPI collective operations [12]. We can define it as a reduction of the vector of values by some defined operation, e.g. sum. In MPI, the reduced values are distributed over the cooperating processes and each process has the vector of the values to be reduced. In the end of the operation, each vector should have the same result, see example in Fig. 1.

Process arrival pattern (PAP) [4] describes a behavior of process arrivals into a collective operation. Formally it is a tuple $(a_0, a_1, \ldots a_{P-1})$, where is $a_i$ is the $i$-th process arrival time (PAT) and $P$ is the number of processes participating in the operation. Similarly we can define process exit pattern (PEP): $(f_0, f_1, \ldots f_{P-1})$, where $f_i$ is the time when process $i$ finished the collective operation. For the evaluation purposes we propose to use the *average elapsed time*: $\bar{e} = \sum_{i=0}^{P-1} \frac{f_i - a_i}{P}$.

We can perceive $\bar{e}$ as the mean time, which each process spent in the collective operation, thus we can assume that its minimization causes execution time of the whole application to be decreased. Thus, the average elapsed time can be used to compare the tested all-reduce algorithms and to asses their performance.

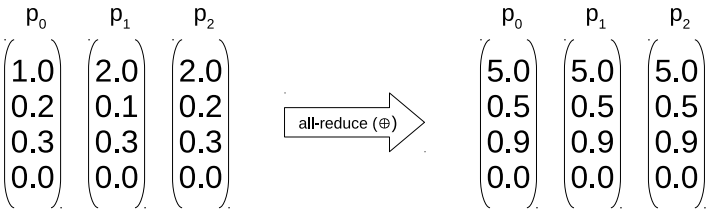


**Fig. 1** Example of all-reduce collective execution for 3 processes ($P = 3$), vector size 4 ($N = 4$) and sum reduction operation ($\oplus$)

Often, the times of process arrivals are not equal for all processes. When these differences are large, we can say that there occurs an imbalanced PAP, formally we can define it, as a PAP with an imbalance factor (a ratio between the highest difference between the arrival times of the processes and time of the simple (point-to-point) message delivery between each other) larger than 1. Such phenomena are ubiquitous in many modern parallel system and it is hard to prevent them [4].

In this paper we investigate the behavior of a set of all-reduce algorithms in a geographically distributed cluster environment. The following algorithms were tested:

- **Parallel Ring (Ring)** a native OpenMPI algorithm used for the proposed configuration [14],
- **Pre-Reduced Ring (PRR)** a PAP-aware algorithms proposed in [10],
- **Hierarchical reduce-broadcast (HR1)** an intuitive, hierarchical algorithm performing reduce to one root process in each site, then performing all-reduce between site root processes and finally broadcasting results locally within sites,
- **Hierarchical all-reduce-all-reduce (HR2)** a hierarchical algorithm first performing all-reduce locally, and then all-reduce between remote sites without a main root [5],
- **Rabenseifner (Rab)** an algorithm performing binomial tree based reduce-scatter and then, also binomial tree based, all-gather operations [13],
- **Butterfly (Bfly)** a binomial tree based algorithm, performing recursive doubling all-to-all send-receives [14].

## 3 Related works

There are many solutions related to the geographically distributed computations. We would like to emphasize the closest approach related to the perfromed experiments—the direct predecessor of the tested architecture, or even its extended version. The broadly geographical distributed compute cluster was presented in [8]. The InfiniCortex project covered connection of a large set of nodes between seven sites over

the world: A*CRC – Singapore and Stony Brook University – USA, SC15 show floor – USA, Poznan Supercomputing and Networking Centre – Poland, University of Reims Champagne-Ardenne – France, Georgia Tech – USA (Clos3 networked cluster), National Computational Infrastructure – Australia and University of Alberta and Obsidian Research – Canada. Its uniqueness was based on the following concepts: high bandwidth (10/100 Gbps), InfiniBand for intra- and interconnection, and support for heterogeneous network topologies. The solution was tested using a number of scientific workflows, proving empirically its usefulness and high performance.

The study on process arrival patterns (PAPs) in a compute cluster environment was presented in [4]. The analysis of a number of existing applications, which used extensively MPI collective communication, showed an ubiquity of imbalanced PAP occurrence, and their high influence on the overall application performance. The authors proposed a new approach to tune the applications, by selecting a collective operation algorithm considering the PAP occurrence. Their solution was introduced and tested on all-to-all collective operation, and was used with their STAR-MPI self-tuning platform, showing its significant advantage over the regular non-tuned approach. In [9], Patarasuk et al. presented two PAP-aware broadcast algorithms dedicated to large messages, one for the non-blocking model of message-passing and the other for the blocking one. They performed theoretical analysis and experimental research, both showing better performance of the proposed algorithms in comparison to the existing ones. The reduce PAP-aware algorithms were presented in [7] and [6]. The former was used for data vectors requiring atomicity and the latter for the segmentable ones. In both cases the theoretical analysis and experimental results showed advantage of the newer approach. In [10] we proposed two new PAP-aware algorithms for all-reduce collective operation. The former extended typical linear tree approach (SLT, Sorted Linear Tree), while the latter was based on parallel ring (PRR, Pre-Reduced Ring). In the experiments, PRR showed better performance in comparison to SLT as well as the regular algorithms used in typical MPI implementations, and was selected to be used for the current research. Finally, in [11] we presented a collection of eight PAP-aware algorithms realizing scatter and gather operations. They were based on sorted linear and binomial trees, executed with and without participation of an additional communication thread. Similarly to the previous results, the experiments showed the advantage of the proposed approach in imbalanced PAPs' environment.

The reduce and all-reduce two-level hierarchical algorithms were presented in [5]. The authors performed in-depth analysis of the communication complexity of the algorithms as well as executed a set of tests on a real supercomputer. The experiments utilized a local (without geographical distribution) cluster with an assumption of balanced PAPs' environment. For the specific configuration, the proposed algorithms showed their advantage over other state-of-the-art solutions.

## 4 Testbed environment

The utilized distributed environment was set up between two supercomputer centers in Poland: Centre of Informatics - Tricity Academic Supercomputer & networK (CI TASK), at Gdansk University of Technology and Interdisciplinary Centre for Mathematical and Computational Modelling at University of Warsaw, see Fig. 2. These two sites are located geographically 300 km from each other, however the used optical fiber circuits configuration spans a much longer distance: 900 km, forcing harder conditions of the transmission including larger latency.

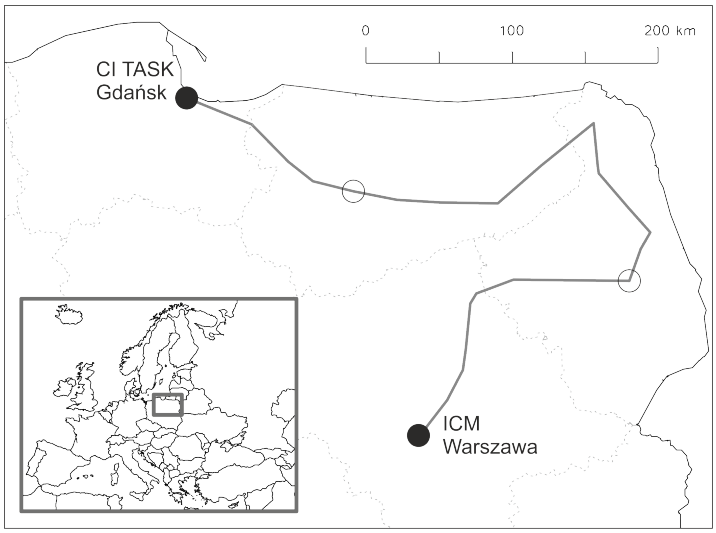


**Fig. 2** Geographical localization and the interconnection of the distributed cluster

The testbed, see Fig. 3, included two cluster environments in ICM and CI TASK, 10 nodes each. Server nodes used in ICM were HUAWEI RH-1288 v3, 2x Intel E5-2680 v3, 4x 6TB SATA drives, whereas nodes used in CI TASK were HPE ProLiant XL230a Gen9, 2x Intel E5-2670 v3, diskless. InfiniBand interconnect in both clusters was based on Mellanox SX6025—InfiniBand switching system with 36 (FDR) 56 Gbps ports and 4 Tbps aggregate switching capacity. Each server was equipped with Mellanox FDR (56 Gbps) Connect-X3 interface card used for the InfiniBand link and 1 GE link for management.

InfiniBand Extenders used in testbed (IBEX G40 - QDR InfiniBand RDMA based Extension Platform) were equipped with one QDR InfiniBand interface and one 40 GE port. IBEX G40 is a 1 U rack unit with a power consumption of less than 140 W. Total buffer capacity allows extending InfiniBand connection up to 15 000 km. The 100 GE circuit spanned between Warsaw and Gdansk was provided by Polish National Research and Education Network (PIONIER) in cooperation

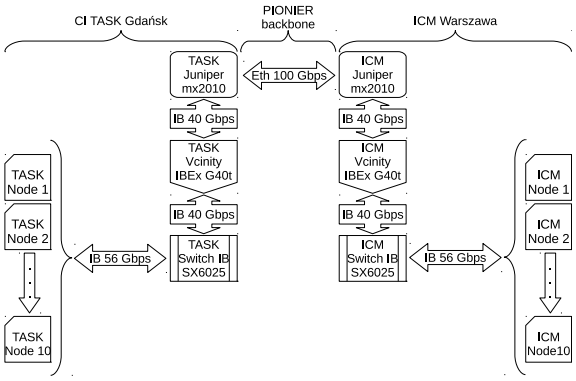


**Fig. 3** Network architecture of the distributed cluster

with Poznan Supercomputing and Networking Centre. The 900 km long circuit introduces 9 ms RTT latency.

We can notice, that the above network architecture is asymmetrical, in the sense that the communication between the local nodes is much faster than between the remote ones. Fig. 4 presents the measured transfer times of the simple one-to-one MPI send-receive operations. We can observe that the tilt for the both cases is similar, meaning the bandwidth is approximately the same, however the offset, over 13 ms (with MPI and other overheads), shows the high latency for remote communication.

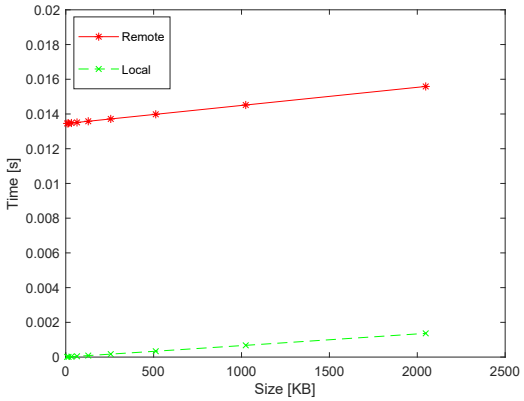


**Fig. 4** Time measurements of sending plain message between two nodes between the remote sites and locally

## 5 The experiments

The experiments were performed in the distributed cluster using a mini-benchmark testing the described all-reduce algorithms (see the list in section 2). The configu-

ration covered both cases, when process arrival times are equal as well as when the PAPs are imbalanced.

## *5.1 Mini-benchmark description*

For the test purposes we used a mini-benchmark proposed in [10] with an extended set of algorithms (see the list in section 2). The code is written in C and the inter-node communication is supported by the typical MPI [2] implementation: Open-MPI [1]. The measurements are performed using the `MPI_Wtime()` function, and the synchronization between the processes is ensured by double invocation of the `MPI_Barrier()` function before every test.

Each test was executed repeatedly: 128–256 times, the exact number depends on the size of the reduced data. The data are generated randomly, so are the process arrival times (PATs), using uniform distribution with maximum delay ranging from 0 ms (a perfectly balanced PAP) to 500 ms. The processes were distributed equally between the sites, and the their total number varied between 2 and 20.

## *5.2 Results for balanced PAPs*

The results presented in this section were gathered for a typical configuration where all processes start the tested collective operation at the same time, i.e. for perfectly balanced PAPs. This approach is almost always used in collective operation performance assessments provided in the literature. We argue that, it is unrealistic in comparison to a real environment [4], however we provide these values for reference purposes.

Table 1 presents the results of the mini-benchmark execution. We can observe that the default algorithm (parallel ring) is the slowest one, the next one is pre-reduced ring, and it seems that the PAP-aware improvement (in comparison to the parallel ring) helped in the case of the asymmetric connection. Both algorithms have the same communication complexity: $O(P)$, where $P$ is a process/node number.

**Table 1** Mini-benchmark execution results: $\bar{e}$—average elapsed time, speedup—the acceleration of the elapsed time for a given algorithm in comparison to parallel ring (default in OpenMPI). The tests were executed on 20 nodes (10 in CI TASK and 10 in ICM) for data size: 512 K of float numbers

| | Ring | PRR | Rab | Bfly | HR1 | HR2 |
|---|---|---|---|---|---|---|
| $\bar{e}$ [ms] | 468 | 339 | 162 | 91 | 44 | 47 |
| speedup | 1.00 | 1.38 | 2.89 | 5.13 | 10.54 | 9.87 |

The next two algorithms: butterfly and Rabenseifner are based on binary tree data distribution and gathering, thus they seem to take advantage of this feature (communication complexity: $O(\log P)$), and perform much better than the ring based ones. Usually, in the case of InfiniBand, where the latency is low, this does not influence the elapsed times so much (in the terms of bandwidth, the rings are more efficient). However, the usage of the IB-Eth-IB converter, and long-distance transmission increases the latency giving the advantage to the tree based algorithms.

Finally, the most efficient algorithms are the hierarchical ones. They were designed for asymmetric network communication, and their structure is exactly adapted to the network architecture. Thus, although they are built on the default (ring) all-reduce algorithm, they provide the best results in most cases, accelerating the communication over 10 times.

Fig. 5 presents the scalability view of the taken measurements for the given cluster configurations. Again, the hierarchical algorithms seem to provide the best results, showing almost flat chart of the averaged elapsed times for the increasing node number in the whole measured range. The binomial tree based algorithms (butterfly and Rabenseifner) also present only small raise of the communication times with increasing cluster size, even outperforming hierarchical ones for power of 2 process/node numbers, when they are the most efficient (by design). Finally, in comparison to the others, ring based algorithms (parallel ring and pre-reduced ring) seem to be less scalable showing quasi-linear growth of the elapsed times.

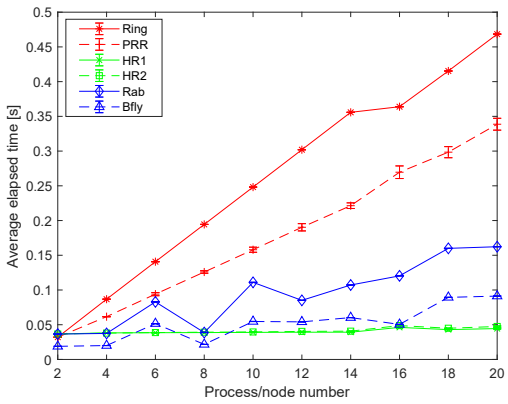


**Fig. 5** Mini-benchmark results showing the scalability of the proposed algorithms. The tests were executed for the data size: 512 K of float numbers, with the equal node numbers in CI TASK and ICM. The error bars are set to $\pm\sigma$ (68% of the measurements for the normal distribution)

Fig. 6 presents the results of the mini-benchmark tests for increasing reduced data size up to 2 M of float numbers, with the constant node set (10 nodes in CI TASK and 10 nodes in ICM). We can observe that for such configuration, the times spent by the processes performing all-reduce operation, are constant, with notable exception of the hierarchical algorithms, which were be able take advantage of the network structure and surpass the other ones.

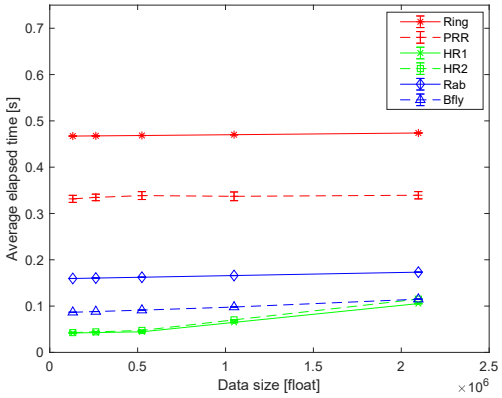


**Fig. 6** Mini-benchmark results showing the impact of reduced data size on average elapsed times of the proposed algorithms. The tests were executed on 20 nodes (10 in CI TASK and 10 in ICM). The error bars are set to $\pm\sigma$ (68% of the measurements for the normal distribution)

We can close the result discussion, with a conclusion that the hierarchical algorithms, because of their design, dedicated for such asymmetrical network architecture, are the best solution for all-reduce collective operation. We should also note that, for the above configuration, where the latency is significant, the binomial tree based algorithms are perform very well. Finally, we can also notice some advantage of the PAP-aware ring algorithm over its regular counterpart for such environment even without the introduction of the imbalanced PAPs.

### *5.3 Results for imbalanced PAPs*

The test results presented in this section extend the previous one by an additional dimension related to the introduction of imbalanced process arrival patterns (PAPs). Often, the measurements are taken in a "sterile" laboratory environment where no additional noise is introduced, which usually is not the case for real HPC computations [4]. Below we analyze the behavior of the tested algorithms for process arrival times (PATs) generated randomly, with unified distribution and maximum delay varying up to 500 ms.

Fig. 7 presents the influence of the increasing maximum delay in randomly generated arrival patterns on the performance of the tested algorithms. We can see that similarly to the balanced PAPs the hierarchical algorithms have the lowest elapsed times, showing the advantage of their design over the other solutions. However we can notice better behavior of the PAP-aware PRR algorithm in comparison to the regular, parallel ring. Also, the binomial tree based algorithms are still in the middle of the ranking.

Fig. 8 presents the comparison of the algorithms in the case of increasing data size in imbalanced PAP environment. We can observe that the tree-based and hierarchical ones provide similar results, and the times are are stable over the range

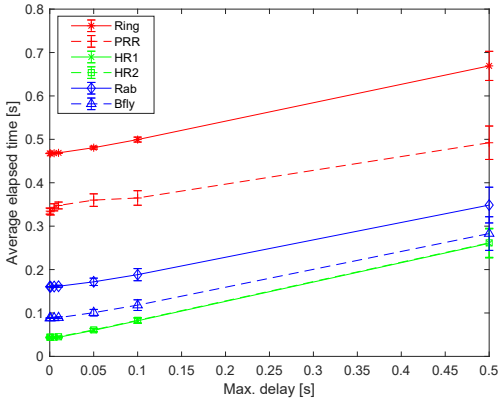


**Fig. 7** Mini-benchmark results showing influence of increasing PAP imbalance on the average elapsed times of the proposed algorithms. The tests were executed for the data size: 512 K of float numbers, using 18 nodes (9 in CI TASK and 9 in ICM). The error bars are set to $\pm\sigma$ (68% of the measurements for the normal distribution)

(with exception of H1 and H2), what can be explained by the high latency and high bandwidth of the inter-site connection.

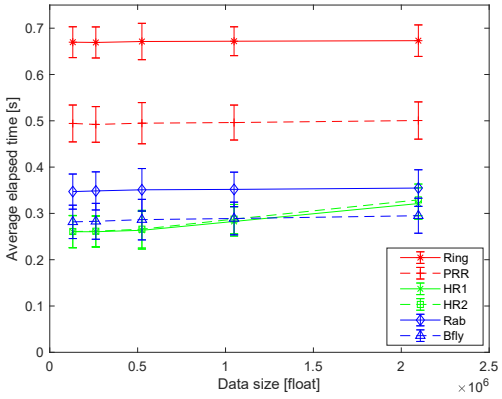


**Fig. 8** Mini-benchmark results showing the impact of reduced data size on average elapsed times of the proposed algorithms in imbalanced PAP environment. The tests were executed for random (uniform distribution) delays: 0–500 ms, using 20 nodes (10 in CI TASK and 10 in ICM). The error bars are set to $\pm\sigma$ (68% of the measurements for the normal distribution)

Fig. 9 presents the scalability of the tested algorithms in the imbalanced PAP environment. We can notice that in this case the hierarchical algorithms behave similarly to the binomial tree based one, showing their advantage over rings. However the results are not so good as for the balanced environment, where the times were almost independent of the process/node number and here we see the growing tendency for all algorithms.

The obtained results showed that for the tested cluster configuration, the PAP imbalances do not overweight the influence of the network architecture, however the PAP-aware PRR algorithm showed some advantage over its regular counterpart: parallel ring. This gives us the clue, that the hierarchical approach with the underlying

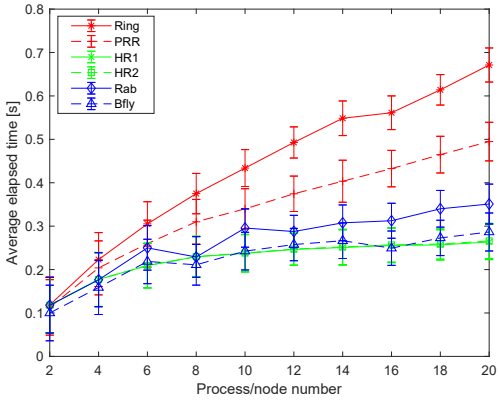


**Fig. 9** Mini-benchmark results showing the scalability of the proposed algorithms in imbalanced PAP environment. The tests were executed for random (uniform distribution) delays: 0–500 ms and for the data size: 512 K of float numbers, with the equal node numbers in CI TASK and ICM. The error bars are set to $\pm\sigma$ (68% of the measurements for the normal distribution)

PAP-aware algorithms, or even a newly designed dedicated PAP-aware algorithm could be an even better solution for the given network configuration.

## 6 Final conclusions and future works

Nowadays, geographically distributed computations are very popular, especially in the cloud environment, where the data centers are distributed over the globe. For HPC, the usual approach was associated with the grid computing, however a rapid development of network technologies enabled usage of RDMA-InfiniBand solutions over the remote sites [8].

In this paper, we presented the results of the behavior of the all-reduce collective operations performed in a geographically distributed InfiniBand cluster. To closely mirror the real-life environment, we introduced an additional dimension related to the imbalanced process arrival patterns (PAPs), where the cooperating processes can arrive to the desired operation in different, non-equal times (PATs).

During the experiments we observed that the best behaving algorithms are based on their hierarchical structure, well-fitting the asymmetrical architecture of the interconnecting network. Moreover, the PAP-aware algorithm (PRR) worked much better than its regular counterpart (parallel ring), however due to the $O(n)$ communication complexity, was surpassed by the ones based on binomial tree, even in imbalanced PAP environment.

The main conclusion of the paper is identification of the need to develop and test new algorithms, which will exploit both the hierarchical structure of the communication and PAP-aware approach to increase the performance. We strongly believe that the further research focused on this area will provide the optimized solution for all-reduce and other collective operations.

Thus, the following activities can be taken in the future works:

- development of new hybrid algorithms combining advantages of the hierarchical and PAP-aware solutions,
- tests of new more efficient, non-Ethernet-in-the-middle solutions, where the long distance optical fiber is used directly by the InfiniBand devices,
- tests of more complex configurations, including 3 and more remote sites,
- simulation and emulation solutions, based on new and existing software, e.g. using MERPSYS [3] package.

The authors anticipate that the development of distributed HPC solutions, as well as ubiquity of imbalanced PAP environment will stimulate the research of new algorithms for all-reduce and other collective operations.

## References

1. Open MPI: Open source high performance computing. URL: https://www.open-mpi.org/. 2017-08-27.
2. The standarization forum for Messsage Passing Interface (MPI). URL: http://mpi-forum.org/. Accessed: 2018-01-24.
3. P. Czarnul, J. Kuchta, M. Matuszek, J. Proficz, P. Rościszewski, M. Wójcik, and J. Szymański. MERPSYS: An environment for simulation of parallel application execution on large scale HPC systems. *Simulation Modelling Practice and Theory*, 77:124–140, 2017.
4. Ahmad Faraj, Pitch Patarasuk, and Xin Yuan. A Study of Process Arrival Patterns for MPI Collective Operations. *International Journal of Parallel Programming*, 36(6):543–570, dec 2008.
5. Khalid Hasanov and Alexey Lastovetsky. Hierarchical redesign of classic MPI reduction algorithms. *The Journal of Supercomputing*, 73(2):713–725, feb 2017.
6. P. Marendic, J. Lemeire, D. Vucinic, and P. Schelkens. A novel MPI reduction algorithm resilient to imbalances in process arrival times. *Journal of Supercomputing*, 72:1973–2013, 2016.
7. Petar Marendić, Jan Lemeire, Tom Haber, Dean Vučinić, and Peter Schelkens. An Investigation into the Performance of Reduction Algorithms under Load Imbalance. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7484 LNCS, pages 439–450. 2012.
8. Marek T Michalewicz, Tan Geok Lian, Lim Seng, Jonathan Low, David Southwell, Jason Gunthorpe, Gabriel Noaje, Dominic Chien, Yves Poppe, Jakub Chrzeszczyk, et al. Infinicortex: present and future invited paper. In *Proceedings of the ACM International Conference on Computing Frontiers*, pages 267–273. ACM, 2016.
9. Pitch Patarasuk and Xin Yuan. Efficient MPI_Bcast across different process arrival patterns. *IPDPS Miami 2008 - Proceedings of the 22nd IEEE International Parallel and Distributed Processing Symposium, Program and CD-ROM*, (1), 2008.
10. Jerzy Proficz. Improving all-reduce collective operations for imbalanced process arrival patterns. *The Journal of Supercomputing*, 74(7):3071–3092, jul 2018.
11. Jerzy Proficz. Process arrival pattern aware algorithms for acceleration of scatter and gather operations. *The Journal of Cluster Computing*, accepted, in press.
12. Rolf Rabenseifner. Automatic MPI counter profiling. In *42nd CUG Conference*, pages 396–405, 2000.
13. Rolf Rabenseifner. Optimization of Collective Reduction Operations. In *Lecture Notes in Computer Science*, volume 3036, pages 1–9. 2004.
14. Rajeev Thakur, Rolf Rabenseifner, and William Gropp. Optimization of Collective Communication Operations in MPICH. *The International Journal of High Performance Computing Applications*, 19(1):49–66, feb 2005.