# Rank Reordering within MPI Communicators to Exploit Deep Hierarchal Architectures of Supercomputers

***Philippe Swartvagher*, Ioannis Vardas, Sascha Hunold, and Jesper Larsson Träff**

*Faculty of Informatics, TU Wien, Austria*

HPC machines are more and more hierarchical: they feature nodes, processors, NUMA domains, L3 caches, ... The performance of applications can depend on how computations and data are mapped to this hierarchy, thus it is important to consider it. Regarding how MPI processes are distributed across available cores, the performance of communications can change, because of the capacity of links on each hierarchy level or contention on these links. Thus, each application has its own optimal process mapping, influenced by the capacities of the hardware and the characteristics of computations and communications. Unfortunately, expressing a mapping based on the hierarchical topology of the system is not straightforward, especially to later execute collective operations [1].

Here, we propose an algorithm to reorder ranks of MPI processes inside a communicator. The idea is to decompose the number corresponding to the MPI rank into a mixed-radix number with a numerical base built from the characteristics of the hierarchy. Changing the order of the base allows to change how the cores in the system hierarchy are enumerated to relabel their ranks. For instance, an order can make two consecutive ranks to always be mapped on different nodes, while another order can make them mapped only on different sockets. Each order is described as a sequence of integers corresponding to the levels of the system hierarchy. For instance, the default order commonly used to enumerate MPI processes is $3-2-1-0$ for a system with a hierarchy composed of 4 levels, such as Node − Socket − NUMA − Core.

These different mappings can have a large effect on overall communication performance. In our experiments, we use this *reordering* method to create sub-communicators. The used order allows to control how MPI processes belonging to a communicator are spread all over the available nodes or rather packed as much as possible inside a single level of the hierarchy. As depicted in Fig. 1 and Fig. 2, results of microbenchmarks executing `MPI_Alltoall` operations in sub-communicators show a performance difference of factor 2 between the best and the worst order when only one communicator executes the collective operation, and a factor of 4 when all communicators simultaneously execute it, respectively. When only one sub-communicator executes collective operations, it is better to have a spread mapping to maximize available bandwidth; while a packed mapping is better when all sub-communicators simultaneously execute collective operations, to minimize contention. By changing the order of rank numbering, we also see an impact on the performance of proxy-applications which execute collective operations in sub-communicators.

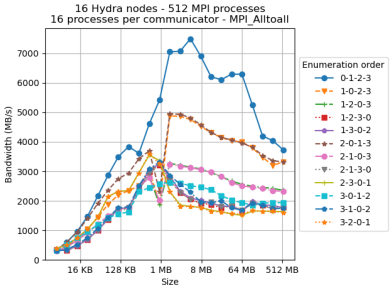


**Fig. 1:** Impact of different reorderings on communication bandwidth when executing `MPI_Alltoall` operation in one communicator of 16 processes (out of 512).

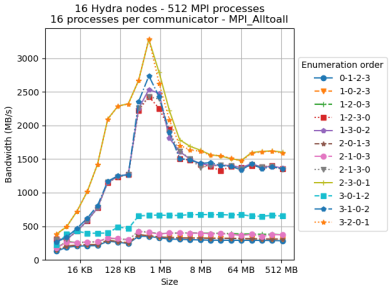


**Fig. 2:** Impact of different reorderings on communication bandwidth when simultaneously executing `MPI_Alltoall` operations in 16 communicators, each with 16 processes.

## References

[1] Mirsadeghi, S.H. and Afsahi A., Topology-Aware Rank Reordering for MPI Collectives. IEEE International Parallel and Distributed Processing Symposium Workshops (2016).