# On the Importance of Data Quality when Tuning MPI Libraries

Sascha Hunold[a] and Alexandra Carpen-Amarie[b]

[a] *TU Wien, Faculty of Informatics, Austria*
[b] *Fraunhofer ITWM, Germany*

**MPI Performance Tuning:** The collective communication operations in MPI provide a standardized way of performing data movements within a group of processes. The efficiency of these collective communication operations depends on the actual algorithm and its implementation. Most MPI libraries provide numerous algorithms for specific collective operations. The problem is that some MPI libraries either do not expose ways to select an algorithm for a specific collective or lack multiple algorithmic options. We show how MPI performance guidelines can help to automatically tune any MPI library if guideline violations occur [1].

**Measuring MPI Collectives using Global Clocks:** An important prerequisite of MPI performance tuning is MPI benchmarking, i.e., the experimental determination of the best performing algorithm. Yet, benchmarking MPI collectives is a complicated tasks, especially if the events under investigation are relatively short. We show how a precise logical, global clock can improve the accuracy when benchmarking MPI functions [3]. We present a hierarchical clock synchronization scheme that distinguishes between intra- and inter-node communication. We further demonstrate that a logical, global clock avoids the influence of `MPI_Barrier` on the benchmark results (cf. Fig. 1). In addition, we propose a new MPI benchmarking scheme called *Round-Time*, which improves the reproducibility of results.

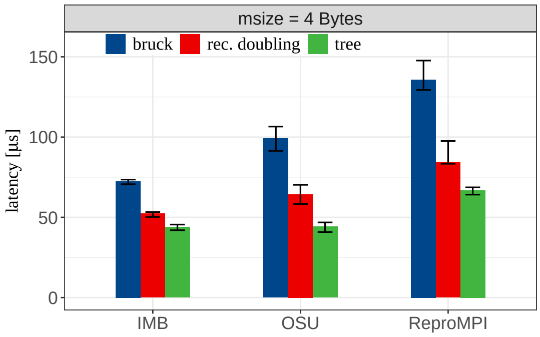

**Fig. 1:** Average latency of `MPI_Allreduce` (reported by various MPI benchmarks) with different `MPI_Barrier` algorithms; Open MPI 3.0.0, 512 = 32×16 processes; machine: jupiter.

**Tuning MPI Collectives with Supervised Learning:** Autotuning tools like `mpitune` determine the best performing algorithm for MPI collective calls. The drawback of these tools is that results can only be applied to cases (e.g., number of processes, message size) for which the tool has performed the optimization. We present another approach, where we create a classifier that takes the collective operation, the message size, and communicator characteristics (number of compute nodes, number of processes per node) as an input and gives the predicted best algorithm as an output [2]. Lastly, we demonstrate that our classification model often outperforms the default configuration of Intel MPI or Open MPI on recent computer clusters (cf. Fig. 2).

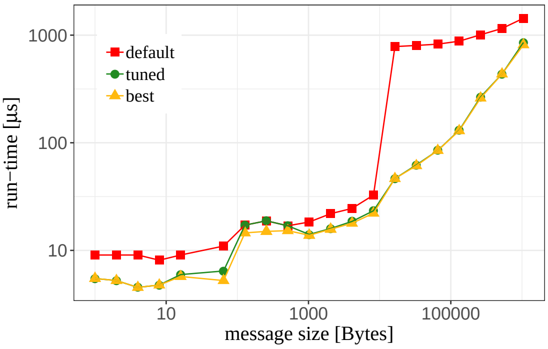

**Fig. 2:** Run-time of `MPI_Bcast` for various message sizes and with 252 = 9 × 28 processes; Intel MPI 2018.5.274; machine: hydra.

## References

[1] Sascha Hunold and Alexandra Carpen-Amarie. Autotuning MPI collectives using performance guidelines. In *HPC Asia*, pages 64–74, ACM, 2018.

[2] Sascha Hunold and Alexandra Carpen-Amarie. Algorithm selection of MPI collectives using machine learning techniques. In *Supercomputing Workshops*, ACM, 2018.

[3] Sascha Hunold and Alexandra Carpen-Amarie. Hierarchical clock synchronization in MPI. In *IEEE CLUSTER*, pages 325–336, IEEE, 2018.