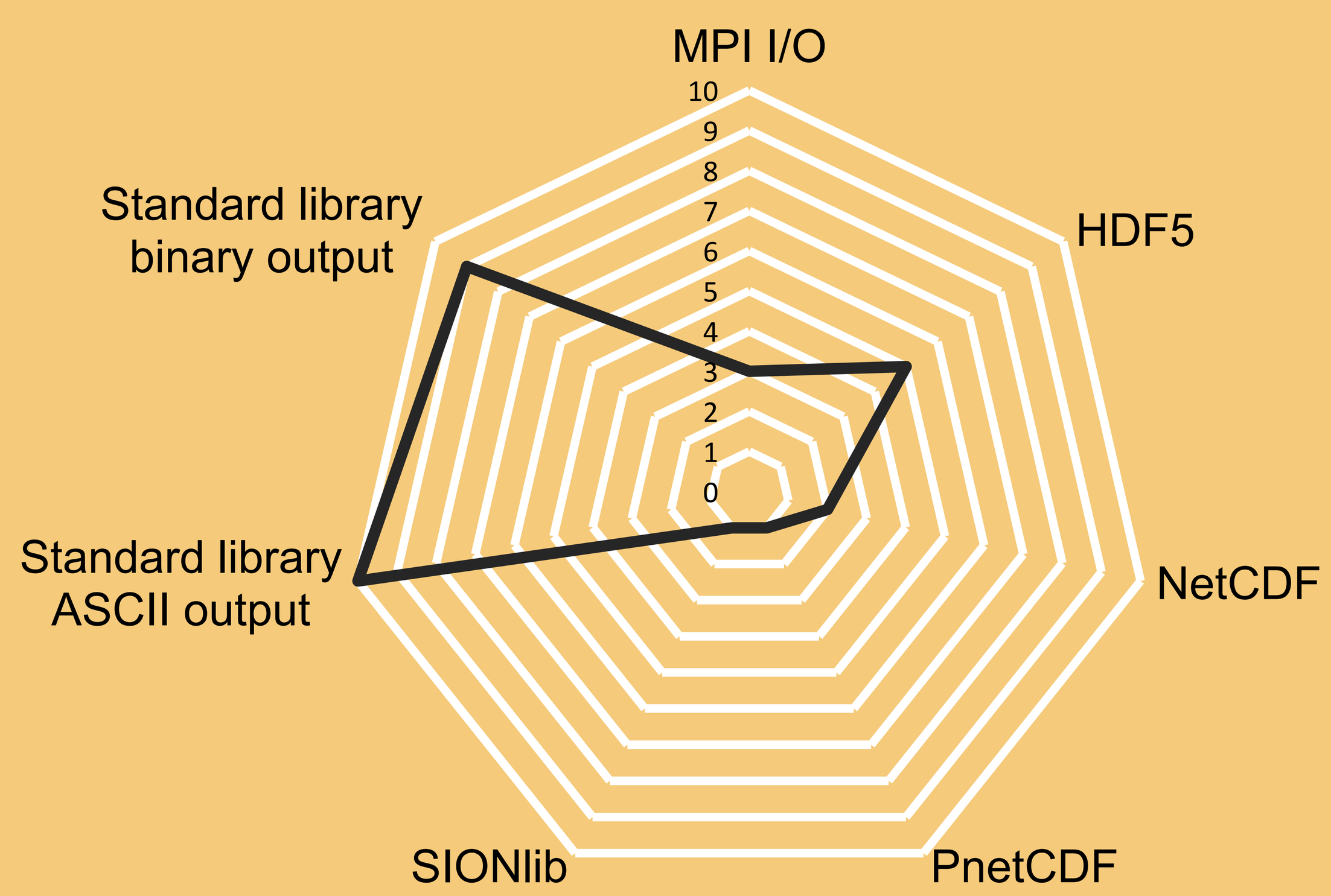
The I/O Benchmarking Activity of the EoCoE Project

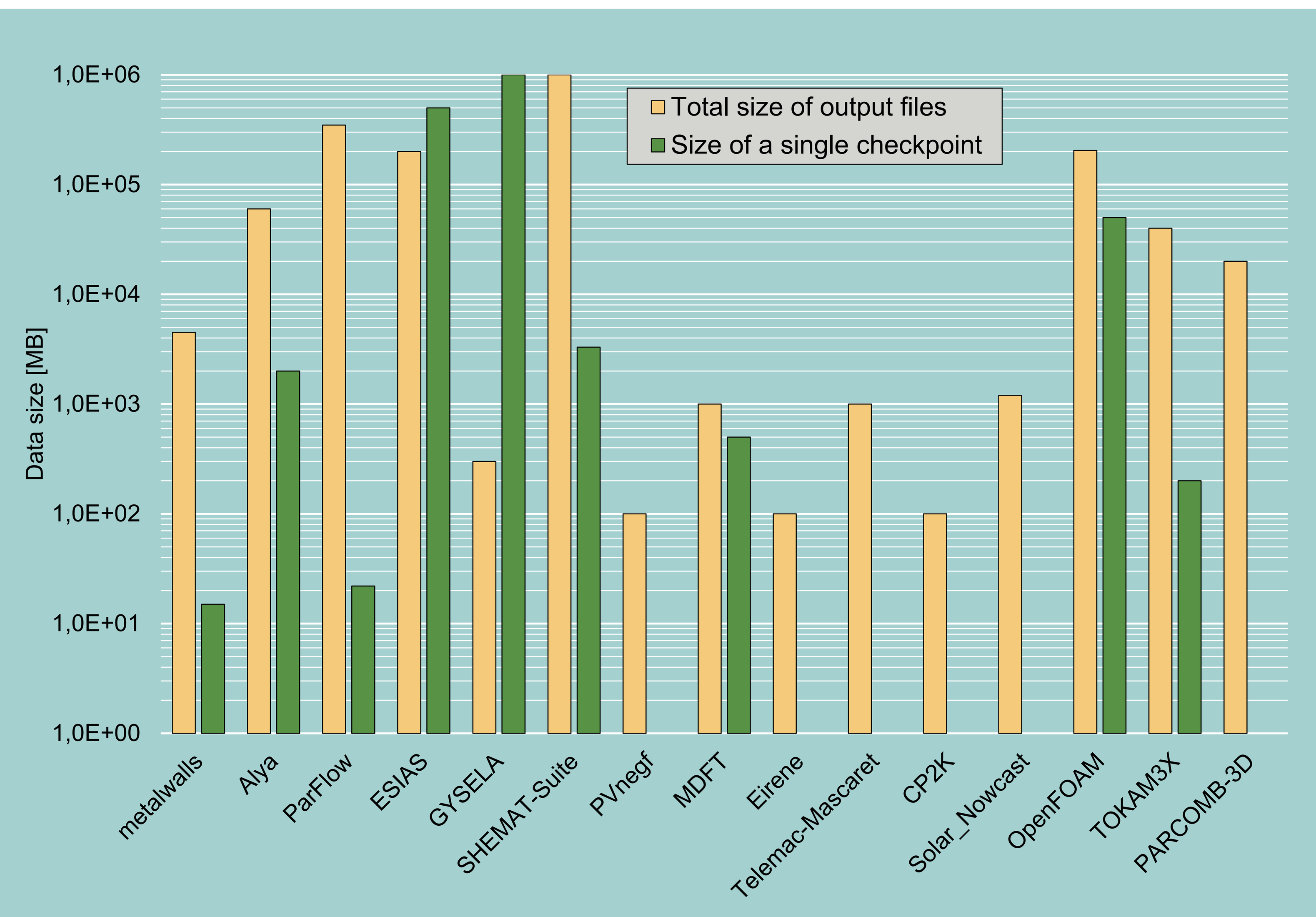
F. Ambrosino[1], M. Brzezniak[2], W. Frings[4], A. Funel[1], G. Guarnieri[1], M. Haefele[3], F. Iannone[1], S. Lührs[4], T.Paluszkiewicz[2], K. Sierocinski[2]
[1]ENEA - Italy, [2]PSNC - Poland, [3]Maison de la simulation - France, [4]Jülich Supercomputing Centre - Germany

Beside the computational scalability of an HPC application, its I/O behaviour can massively influence the overall performance.

To allow **validation and testing of I/O performance bottlenecks and improvements**, an I/O benchmarking activity was established as part of the EoCoE project.

EoCoE applications I/O library distribution





I/O behaviour of the EoCoE applications

An **I/O questionnaire** was designed and circulated to the different EoCoE application owners to gather relevant I/O information and patterns in the context of a typical production run.

I/O benchmarking

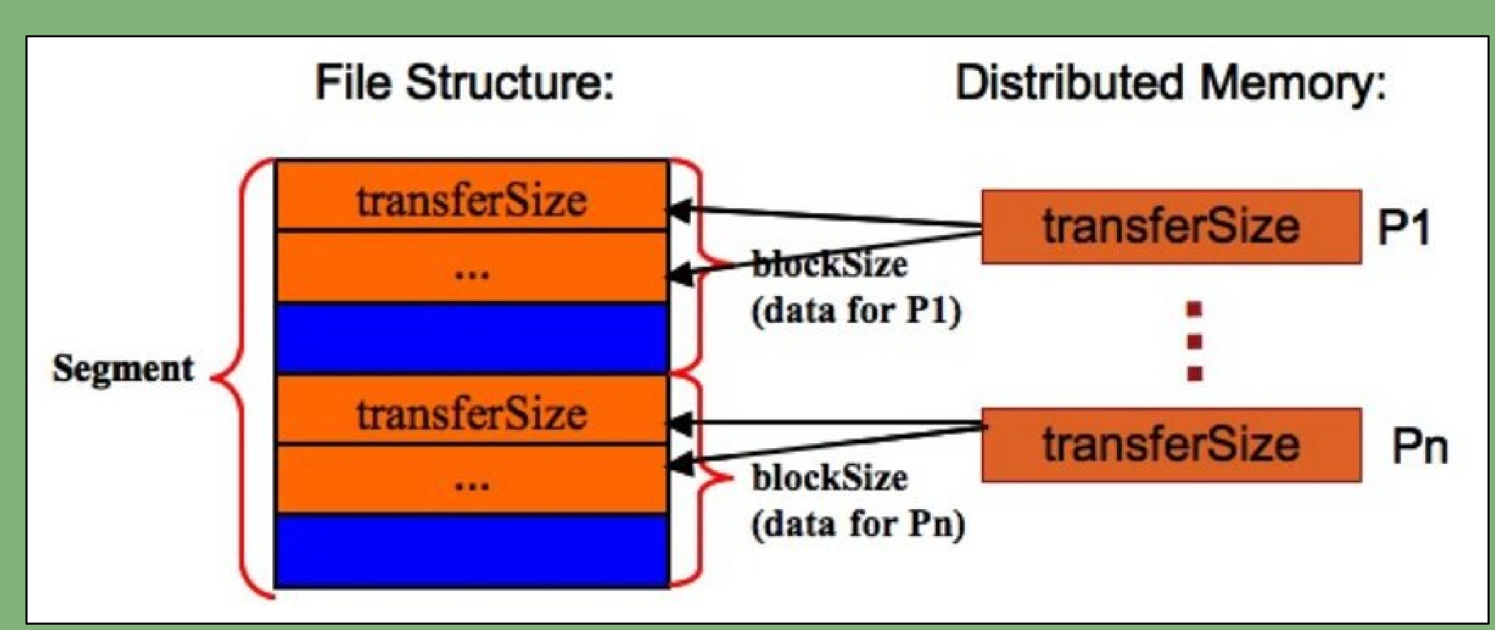
Two benchmark applications - IOR and Partest - were selected to evaluate different types of I/O. Three different systems at PSNC (Eagle), ENEA (CRESCO) and JUELICH (JURECA) are used to run the benchmarks. They were executed through the JUBE benchmarking environment.

Partest

- Benchmark is part of the SIONlib I/O library
- Allow comparison of shared and distributed file I/O
- Supports SIONlib and Posix
- Simulation of typical checkpointing behaviour

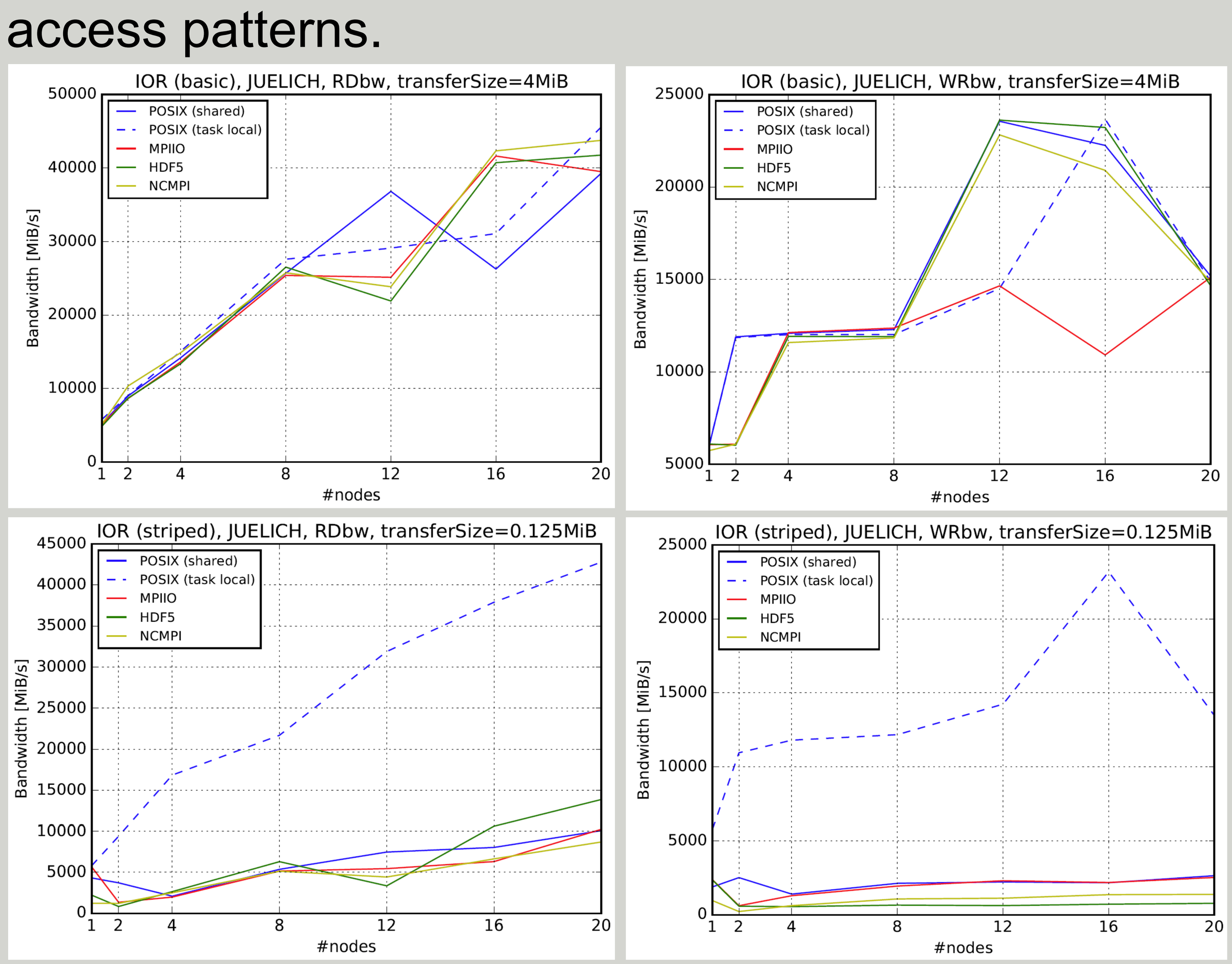
IOR

- Well known and established I/O benchmark
- Initially developed by LLNL (https://github.com/hpc/ior)
- Supports MPIIO, HDF5, PnetCDF and Posix
- Allows to validate library overhead, collective vs. independent I/O behaviour and the dependence of different transfer sizes
- Investigation on compact and strided data layout
- IOR file layout:

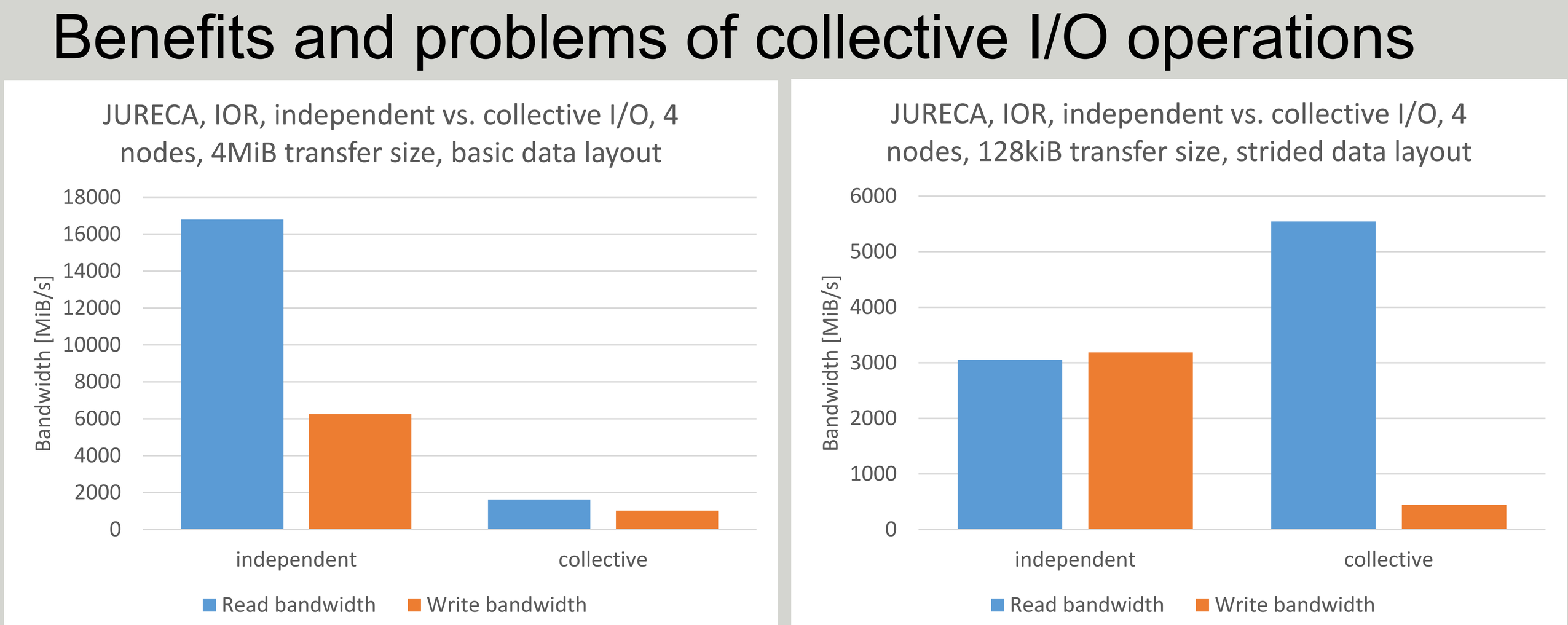


Benchmarking results

Comparison of strided and compact data layouts to visualize the bandwidth difference for different file access patterns.

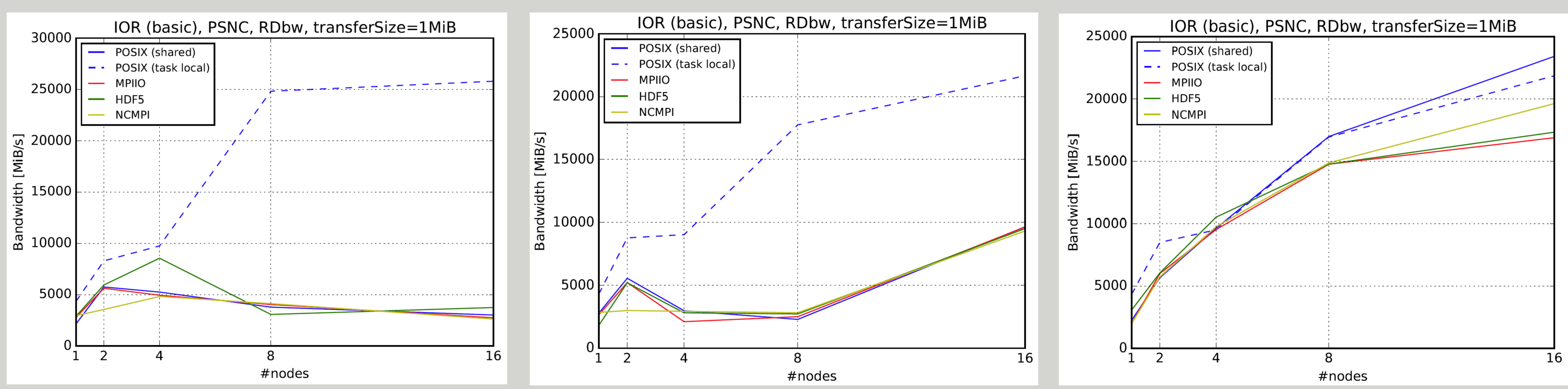


Benefits and problems of collective I/O operations



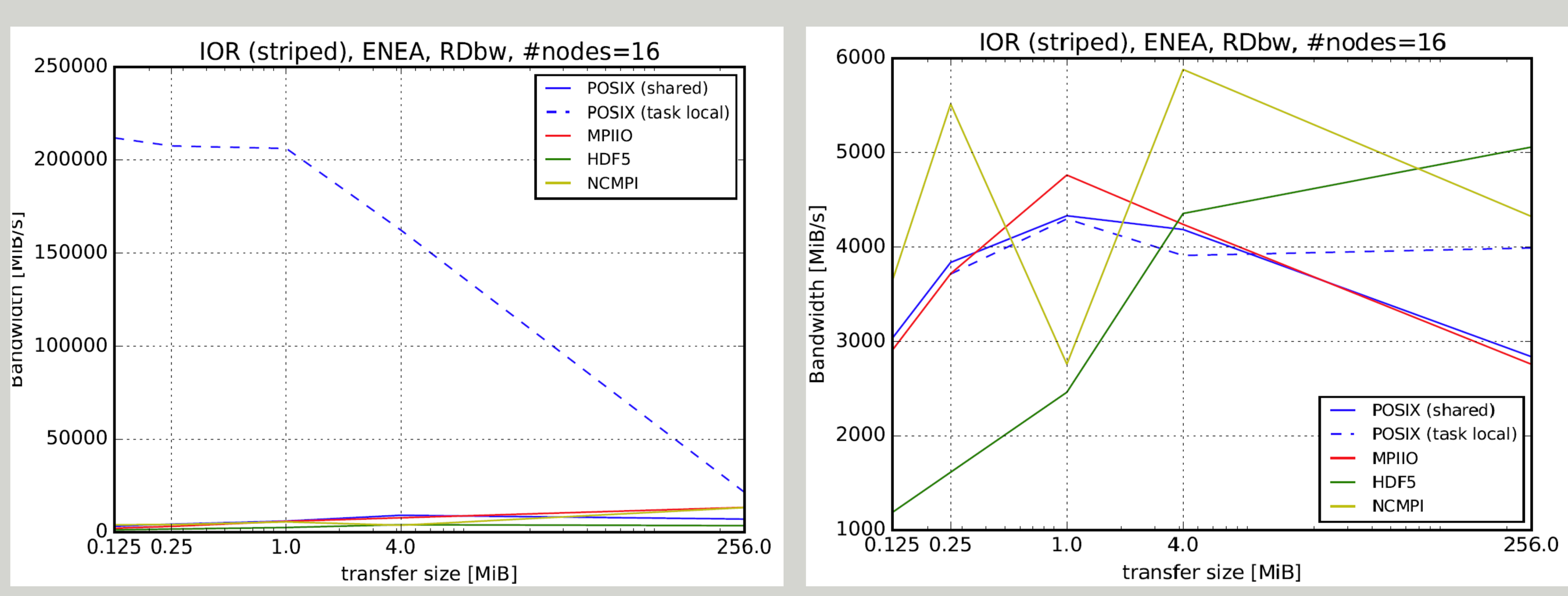
Left: Collective I/O bandwidth degradation for compact file layout; right: Collective I/O read-bandwidth improvement for strided data layout

Lustre based filesystems allow the user to change filesystem parameters such as the number of involved object storage targets (OSTs) and the file stripe size amongst all involved OSTs.



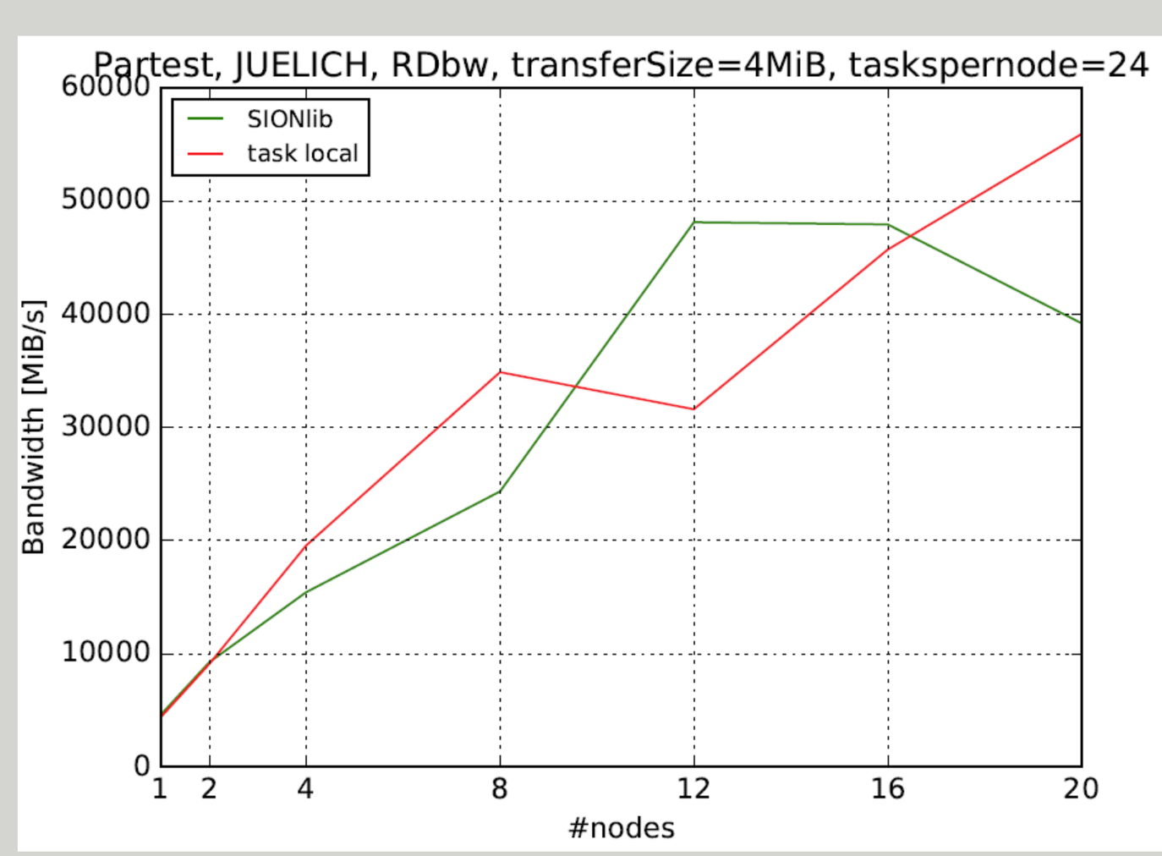
Left to right: (1) Default number of OSTs (12) and default strip-size setting (1MiB); (2) increasing number of OSTs (126); (3) increasing stripe size to align with the individual amount of data per process (256MiB)

Influence of local cache effects



Left: Cache effect in the task local access scheme; right: Avoiding local cache effects by reordering the tasks to use different tasks for the reading than for writing

Comparison of the task local file access scheme, using multiple files or the SIONlib library



European Commission
Horizon 2020 European Union funding for Research & Innovation

This work was supported by the Energy oriented Centre of Excellence (EoCoE), grant agreement number 676629, funded within the Horizon2020 framework of the European Union.