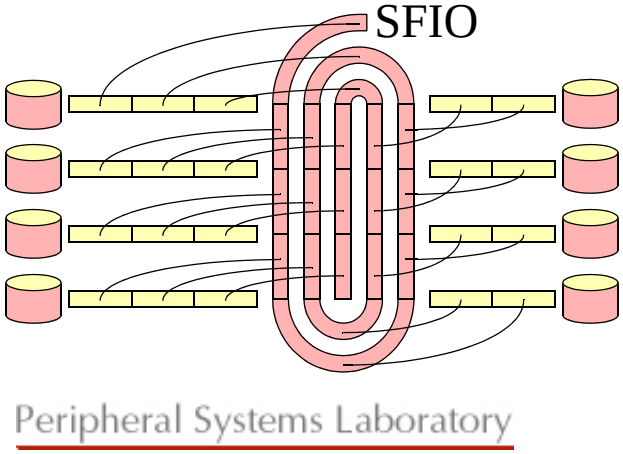
May 23-24, 2001, Sheraton Hyannis, Cape Cod, Hyannis MA

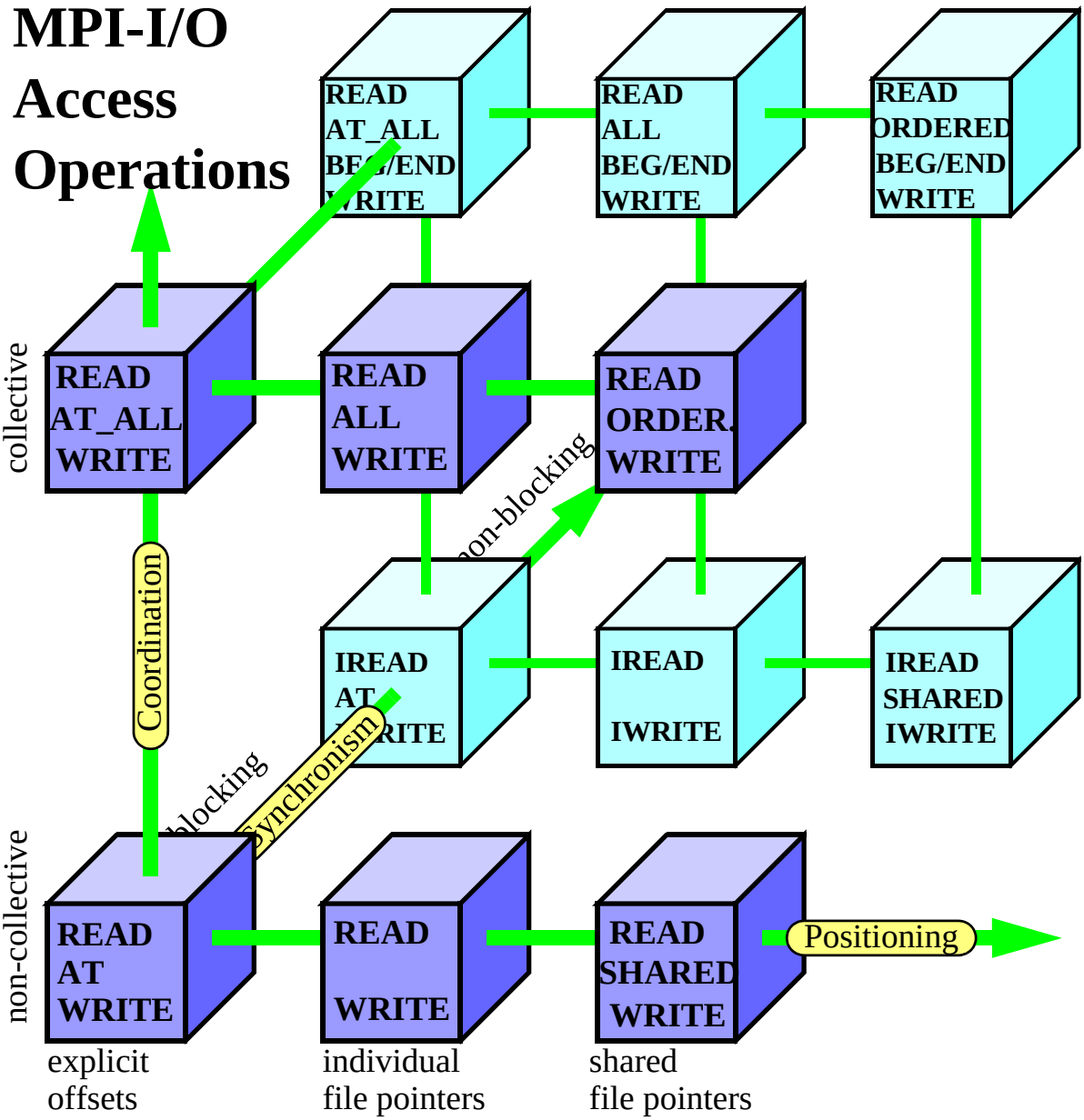
5th Workshop on Distributed Supercomputing: Scalable Cluster Software



Isolated MPI-I/O Solution on top of MPI-1

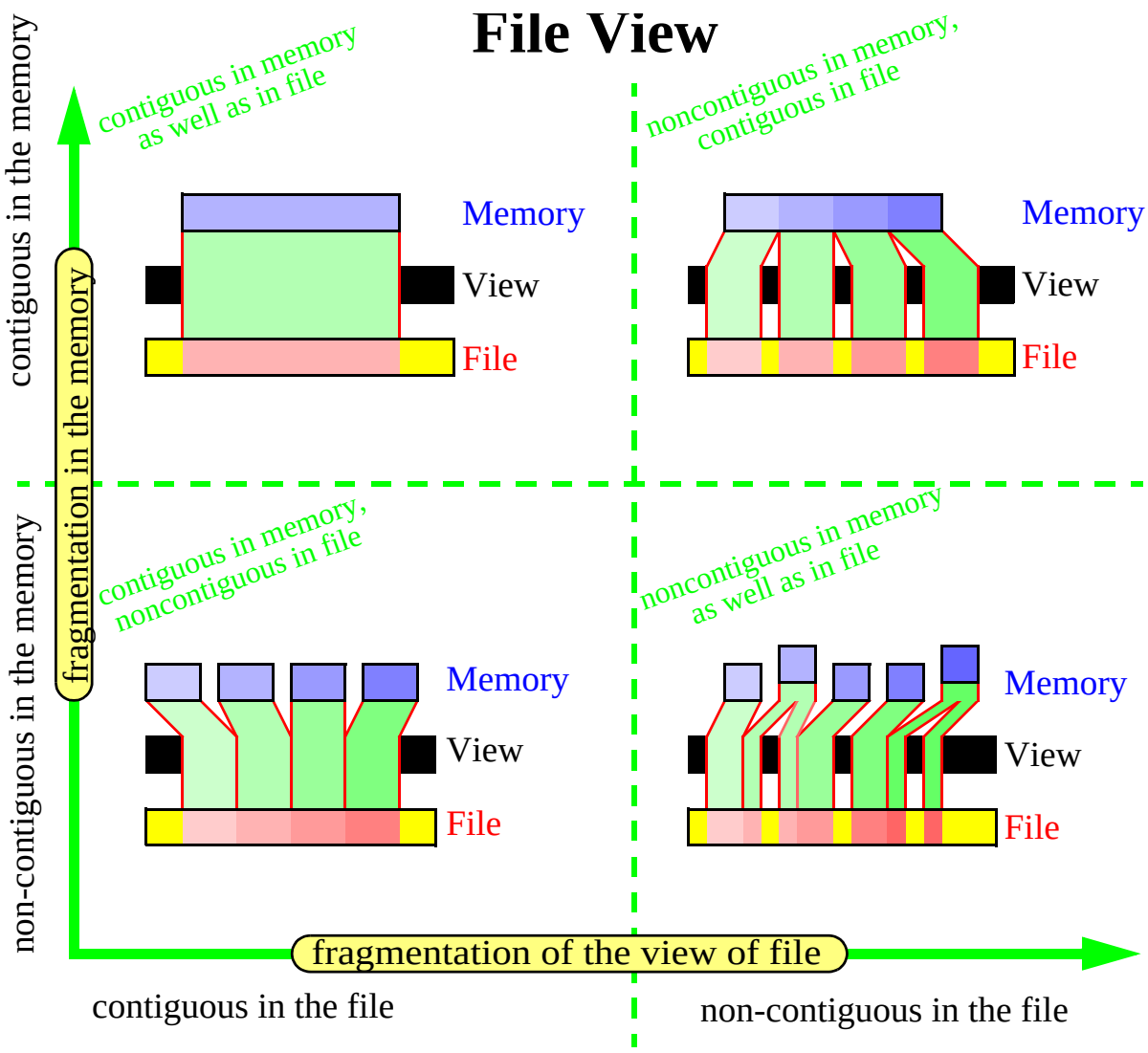
Emin Gabrielyan, Roger D. Hersch
École Polytechnique Fédérale de Lausanne, Switzerland
{Emin.Gabrielyan,RD.Hersch}@epfl.ch

.



The basic set of MPI-I/O interface functions consists of File Manipulation Operations, File View Operations and Data Access Operations. There are three orthogonal aspects to data access: positioning, synchronism, and coordination, and there are 12 respective types of read and of write operations.

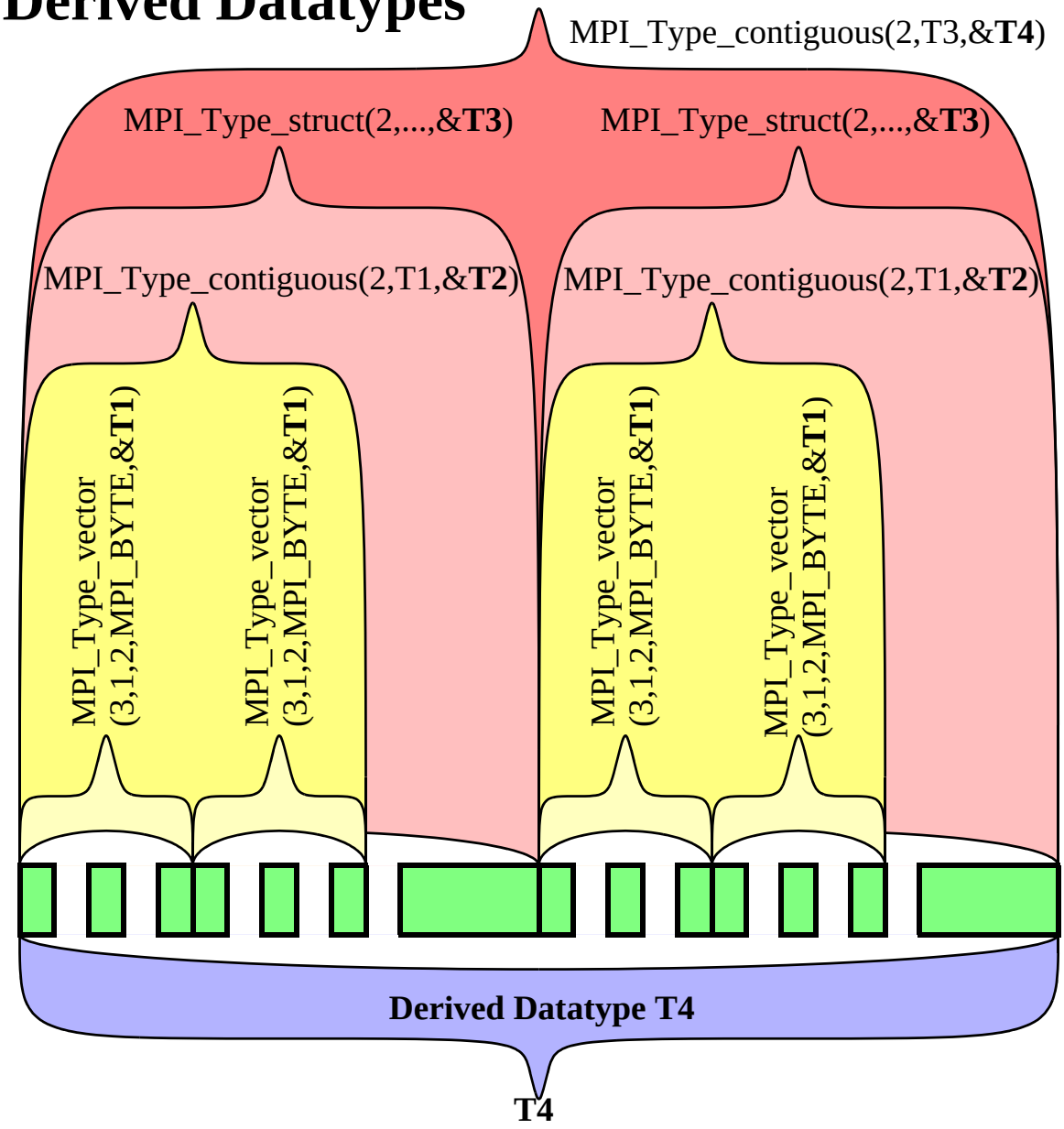
.



The file view is a global concept, which interferes with all data access operations. For each process it specifies its own view of the shared data file: a sequence of pieces in the common data file that are visible for the particular process. In order to specify the file view the user creates a derived datatype, which defines the fragmented structure of the visible part of the file. Since each access operation can use another derived datatype that specifies the fragmentation in memory, there are two additional orthogonal aspects to data access: the fragmentation in the memory and the fragmentation of the file view.

.

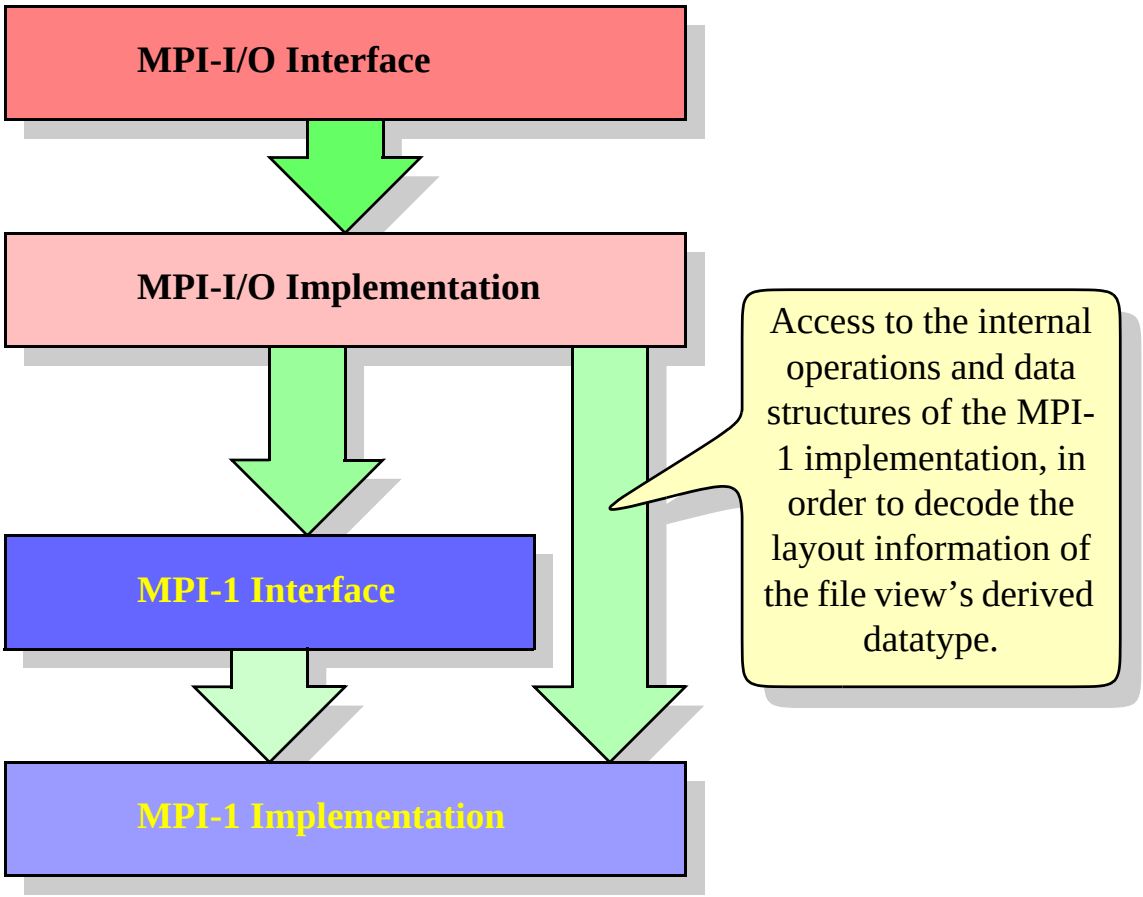
Derived Datatypes



MPI-1 provides techniques for creating datatype objects of arbitrary data layout in memory. The opaque datatype object can be used in various MPI operations, but the layout information, once put in a derived datatype, can not be decoded from the datatype.

.

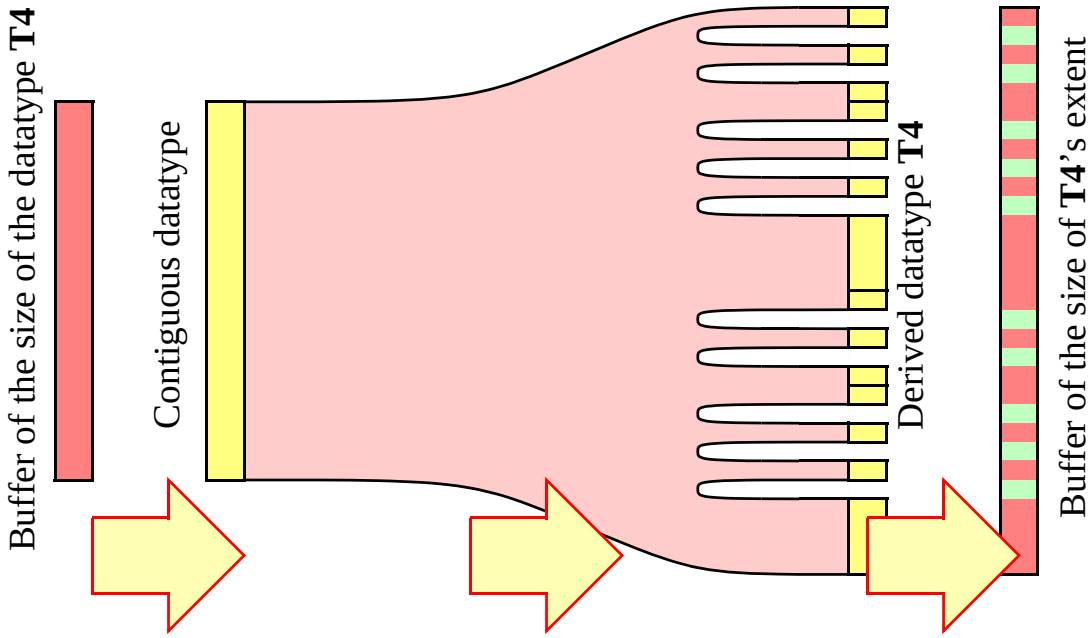
MPI-I/O Implementation



MPI-2 operations and the MPI-I/O subset in particular form an extension to MPI-1. However a developer of MPI-I/O needs access to the source code of the MPI-1 implementation, on top of which he intends to implement MPI-I/O. For each MPI-1 implementation a specific development of MPI-I/O will be required.

.

Reverse Engineering or Memory Painting



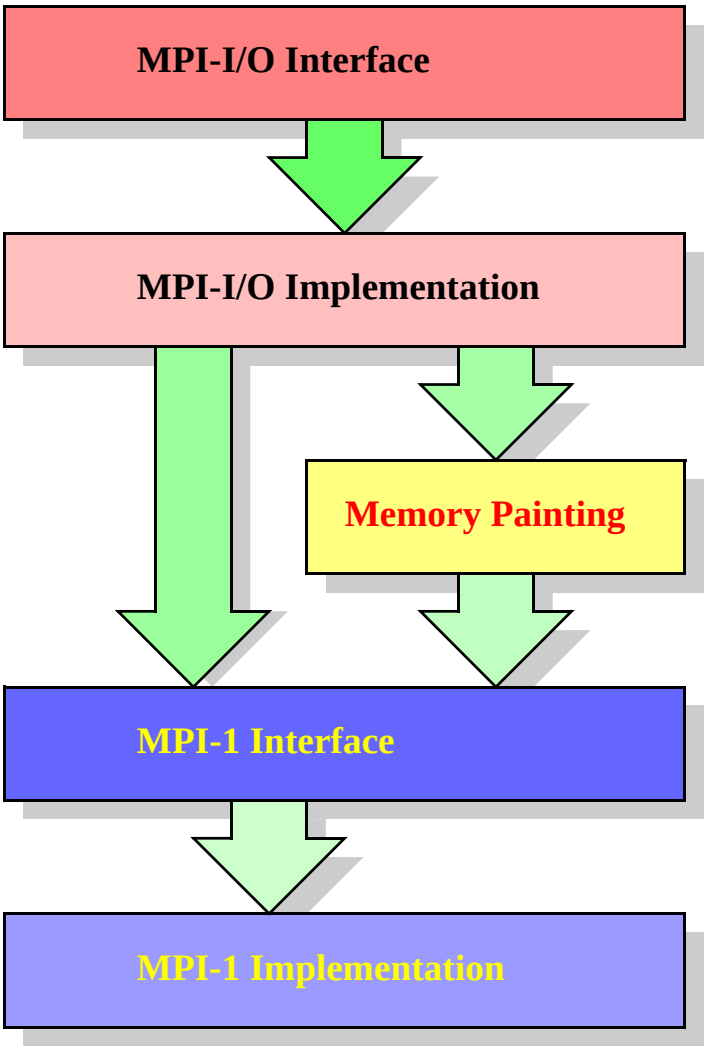
MPI_Send(source,size,MPI_BYTE,...)

MPI_Recv(destination-LB,1,**T4**,...)

The layout information can not be decoded from the datatype, but the behaviour of the datatype depends on the layout. We try to define a special test for a derived datatype, analyse the behaviour of the datatype and based on it, decode the layout information of the datatype. For example, MPI_Recv operation receives a contiguous network stream and distributes it in memory according to the data layout of the datatype. If the memory is previously initialised with a “green colour”, and the network stream has a “red colour”, then analysis of the memory after data reception will give us the necessary information on the data layout hidden in the opaque datatype. In our solution we do not use MPI_Send and MPI_Recv operations, instead we use the MPI_Unpack standard MPI-1 operation to avoid network transfers and multiple processes usage.

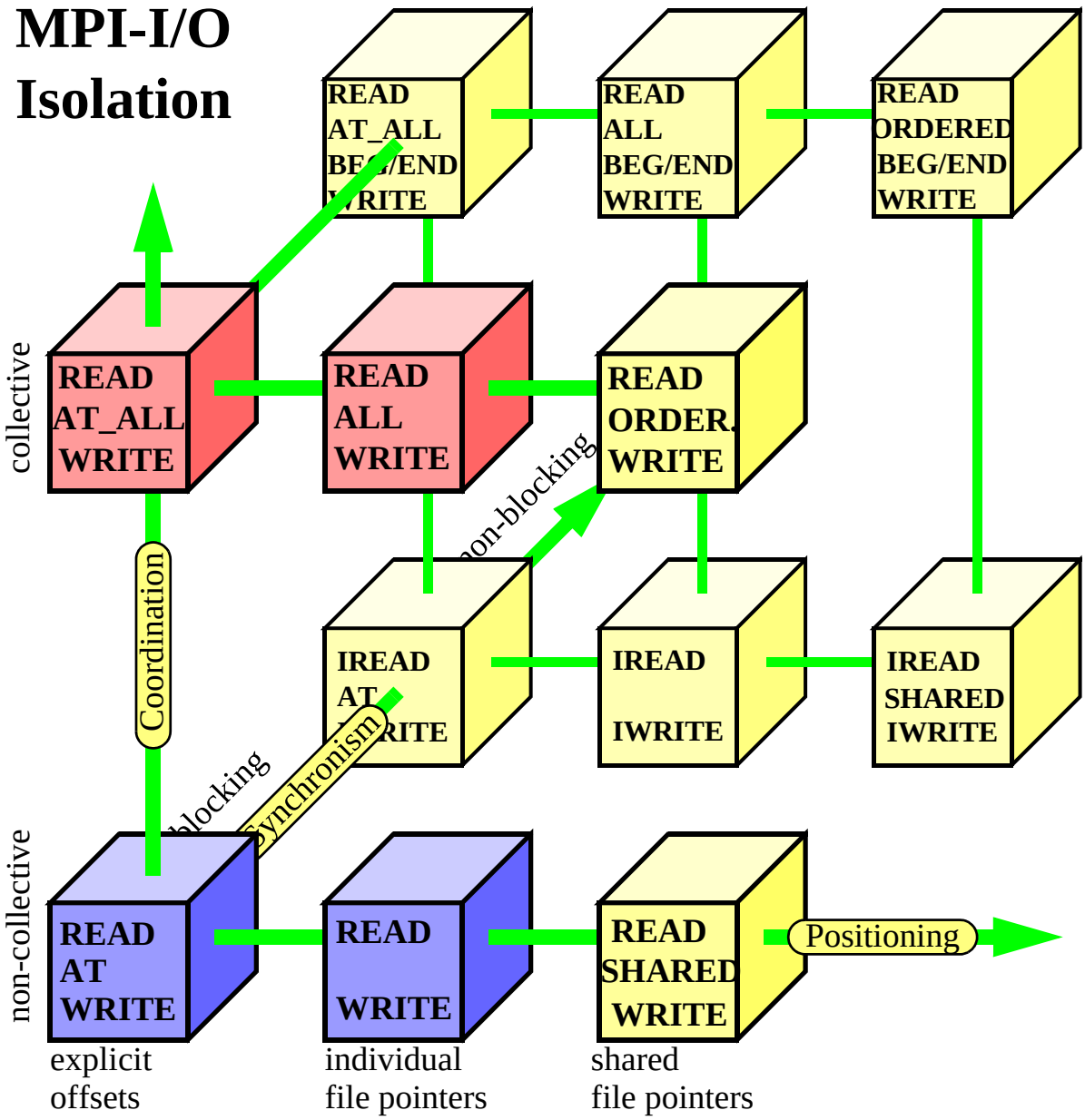
.

Portable MPI-I/O Solution



Once we have a tool for derived datatype decoding, it becomes possible to create an isolated MPI-I/O solution on top of any standard MPI-1. The Argonne National Laboratory's MPICH implementation of MPI-I/O is intensively used with our datatype decoding technique and an isolated solution of a limited subset of MPI-I/O operations has been implemented.

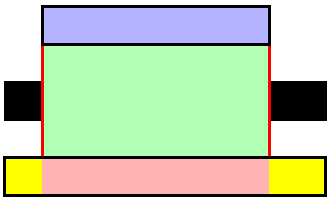
.



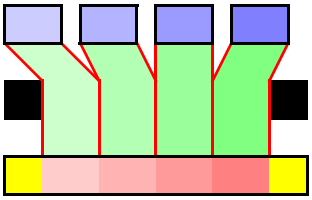
The basic File Manipulation operations MPI_File_open and MPI_File_close; File View operation MPI_File_set_view and blocking non-collective Data Access Operations MPI_File_write, MPI_File_write_at, MPI_File_read, MPI_File_read_at are already successfully implemented in the form of an isolated independent library. Currently we are working on the collective counterparts of blocking operations and trying to make use of the extended two-phase method for accessing sections of out-of-core arrays, on which the ANL implementation is based.

.

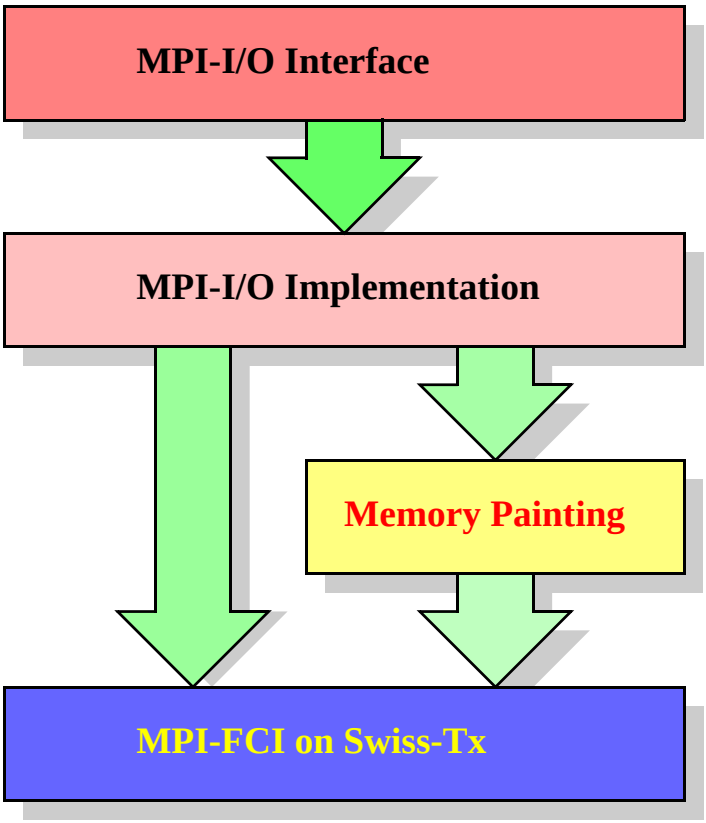
Testing Isolated MPI-I/O

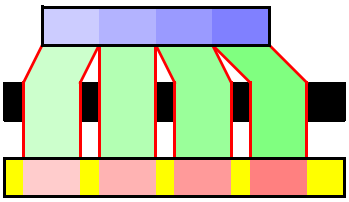


Contiguous memory and file
- MPI_File_write: MPI-FCI **Ok**
- MPI_File_read: MPI-FCI **Ok**
- MPI_File_write_at: MPI-FCI **Ok**
- MPI_File_read_at: MPI-FCI **Ok**

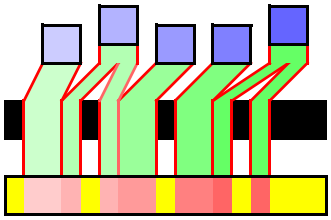


Fragmented memory, contiguous file
- MPI_File_write: MPI-FCI **Ok**
- MPI_File_read: MPI-FCI **Ok**
- MPI_File_write_at: MPI-FCI **Ok**
- MPI_File_read_at: MPI-FCI **Ok**





Contiguous memory, fragmented file
- MPI_File_write: MPI-FCI **Ok**
- MPI_File_read: MPI-FCI **Ok**
- MPI_File_write_at: MPI-FCI **Ok**
- MPI_File_read_at: MPI-FCI **Ok**

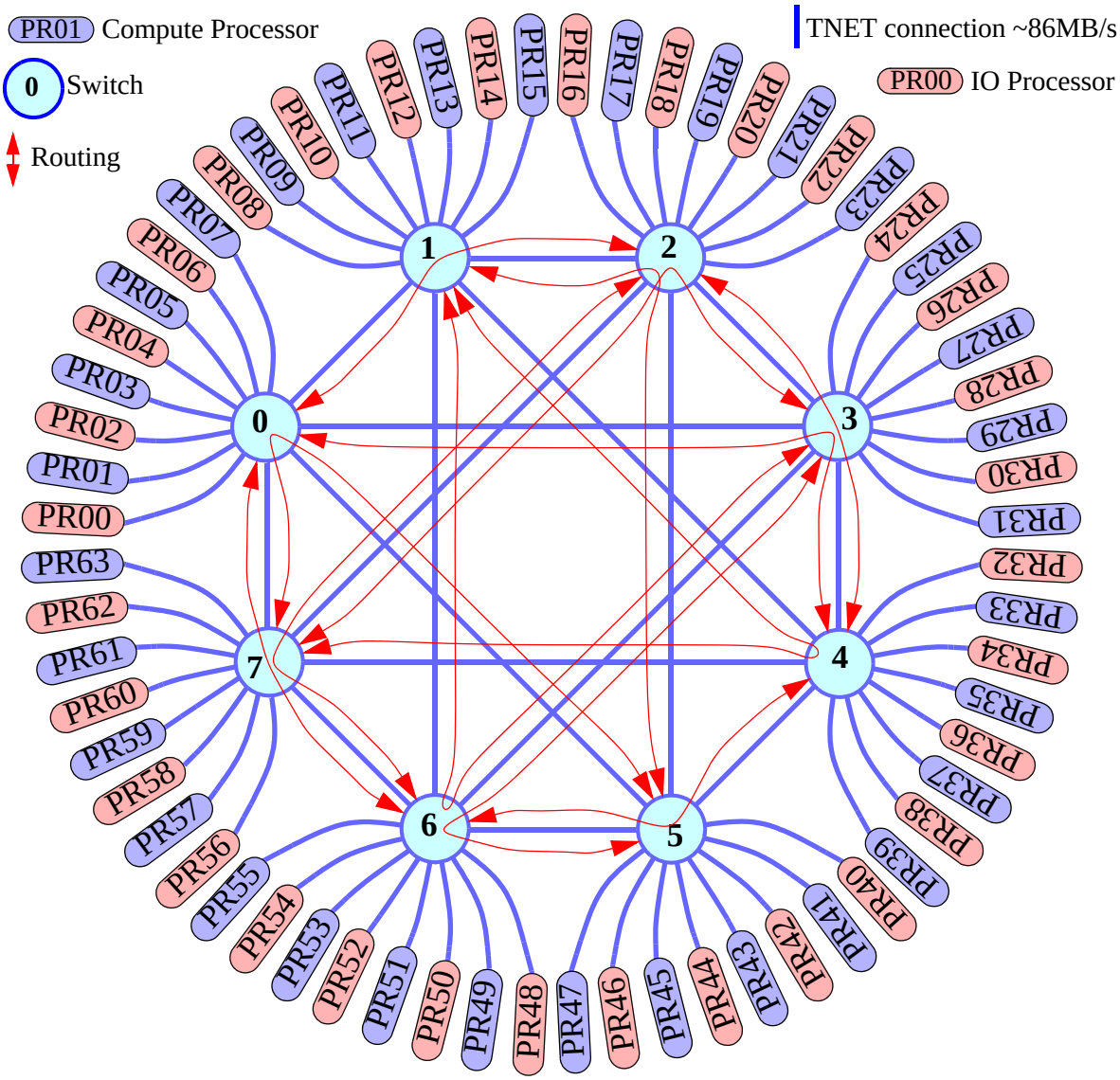


Fragmented memory and file
- MPI_File_write: MPI-FCI **Ok**
- MPI_File_read: MPI-FCI **Ok**
- MPI_File_write_at: MPI-FCI **Ok**
- MPI_File_read_at: MPI-FCI **Ok**

The implemented operations of the isolated solution of MPI-I/O are successfully tested with the MPI-FCI implementation of MPI-1 on the Swiss-Tx supercomputer.

.

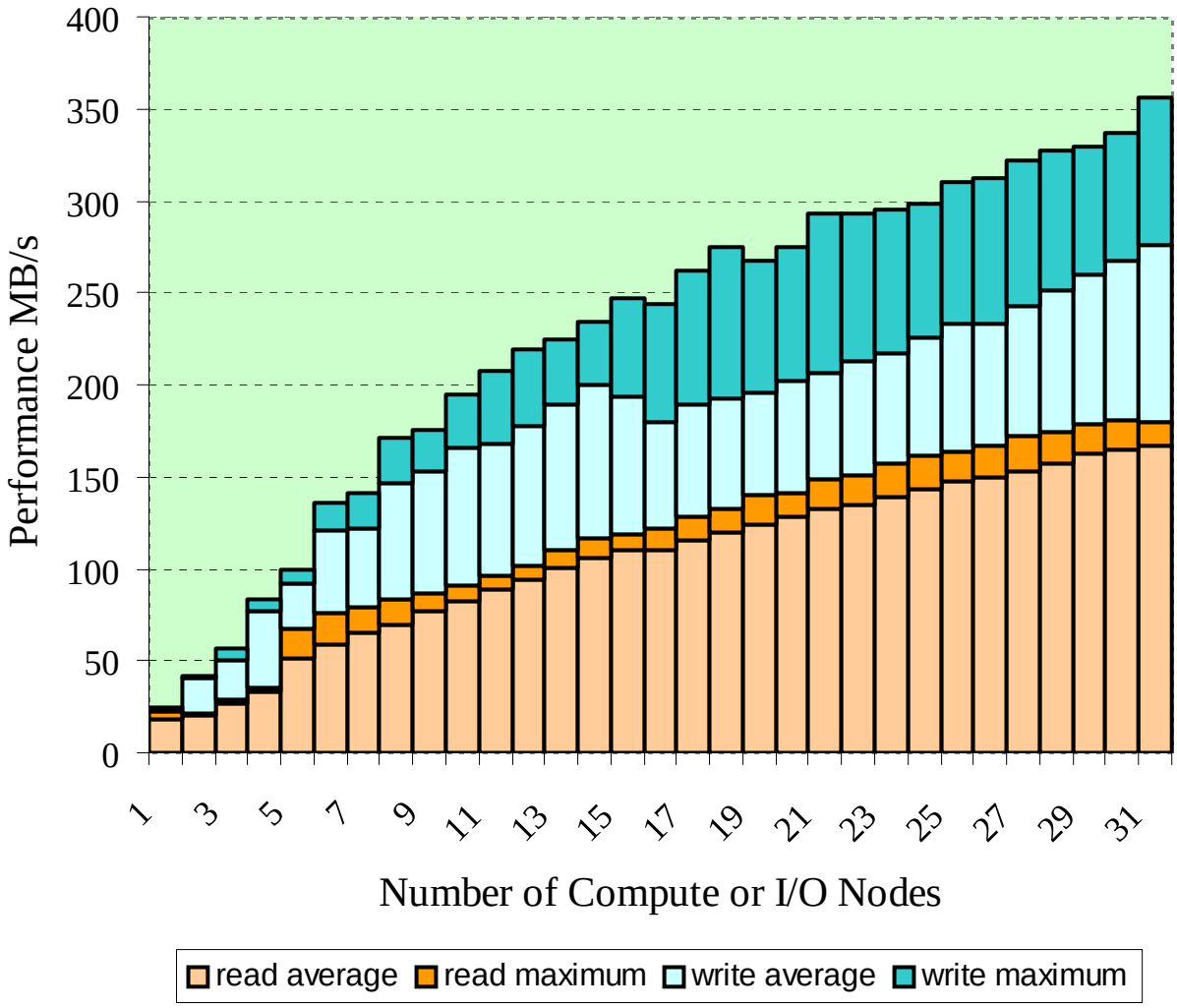
Gateway to the Parallel I/O of the Swiss-T1



At the bottom of the isolated MPI-I/O, we intended to provide as a high performance I/O solution a switching to the Striped File I/O system (SFIO). SFIO communication layer is implemented on top of MPI-1 and therefore SFIO is also portable. We measured a scalable performance of the SFIO on the architecture of the Swiss-Tx supercomputer.

.

SFIO on the Swiss-Tx machine



The performance of SFIO is measured for concurrent access from all compute nodes to all I/O nodes. In order to limit operating system caching effects, the total size of the striped file linearly increases with the number of I/O nodes up to 32GB. The stripe unit size is 200 bytes. The application's I/O performance is measured as a function of the number of Compute and I/O nodes.

.

Conclusion

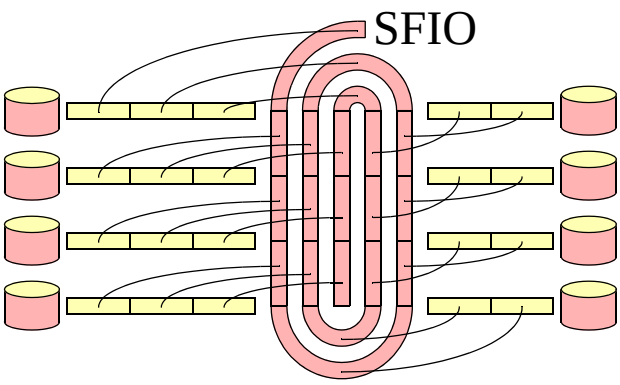
Isolated solution automatically gives to every MPI-1 owner an MPI-I/O, without any requirements of changing, modifying, or specifically interfering to his current MPI-1 implementation.

Future work

- Implementation of blocking collective file access operations.
- Implementation of non-blocking file access operations.
- The remaining File Manipulation Operations.
- Switching to SFIO.

Thank You !

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE



Peripheral Systems Laboratory

.