

MPI-IO: A Retrospective

Rajeev Thakur

25th Anniversary of MPI Workshop
Argonne, IL, Sept 25, 2017

Outline

- Origins of MPI-IO
- How it became part of MPI-2
- What is in MPI-IO
- Which features of MPI-IO turned out to be useful, and which were less useful
- Current status and future

The Origins (1)

- MPI-IO originated in an effort at IBM Research in 1994 to explore the use of the (then) new MPI concepts to define an interface for parallel I/O^[1]
 - At that time, there was a need for a standard parallel I/O interface, just as a few years earlier there was a need for a standard message-passing interface
 - There were many I/O interfaces in existence: Each vendor, parallel file system, or research I/O library had its own interface
1. Jean-Pierre Prost, Marc Snir, Peter Corbett, and Dror Feitelson, “MPI-IO, a message-passing interface for concurrent I/O,” Tech. Rep. RC 19712 (87394), IBM T.J. Watson Research Center, August 1994

The Origins (2)

- MPI seemed like a good basis to build an I/O interface because of similarities between message passing and I/O
 - Writing to a file is like sending a message; reading is like receiving
 - Any parallel I/O API would need
 - collective operations
 - nonblocking operations
 - ability to describe noncontiguous layouts in memory and file
 - separation of application-level messages from I/O-related messages
 - i.e., features that MPI already had

The Origins (3)

- The IBM effort expanded to a collaboration with NASA Ames
- A revised version of the technical report was prepared and distributed at a BoF at SC'94 in Washington DC
 - Peter Corbett, Dror Feitelson, Yarsun Hsu, Jean-Pierre Prost, Marc Snir, Sam Fineberg, Bill Nitzberg, Bernard Traversat, and Parkson Wong, “MPI-IO: A parallel file I/O interface for MPI,” Technical Report RC 19841 (87784), IBM T.J. Watson Research Center, November 1994
- At this point, a large email discussion group was formed, with participation from many institutions
- This group, calling itself the “MPI-IO Committee”, pushed the idea further in a series of proposals, culminating in the version 0.5 release of the MPI-IO specification
 - The MPI-IO Committee, “MPI-IO: A parallel file I/O interface for MPI,” Version 0.5, April 1996.

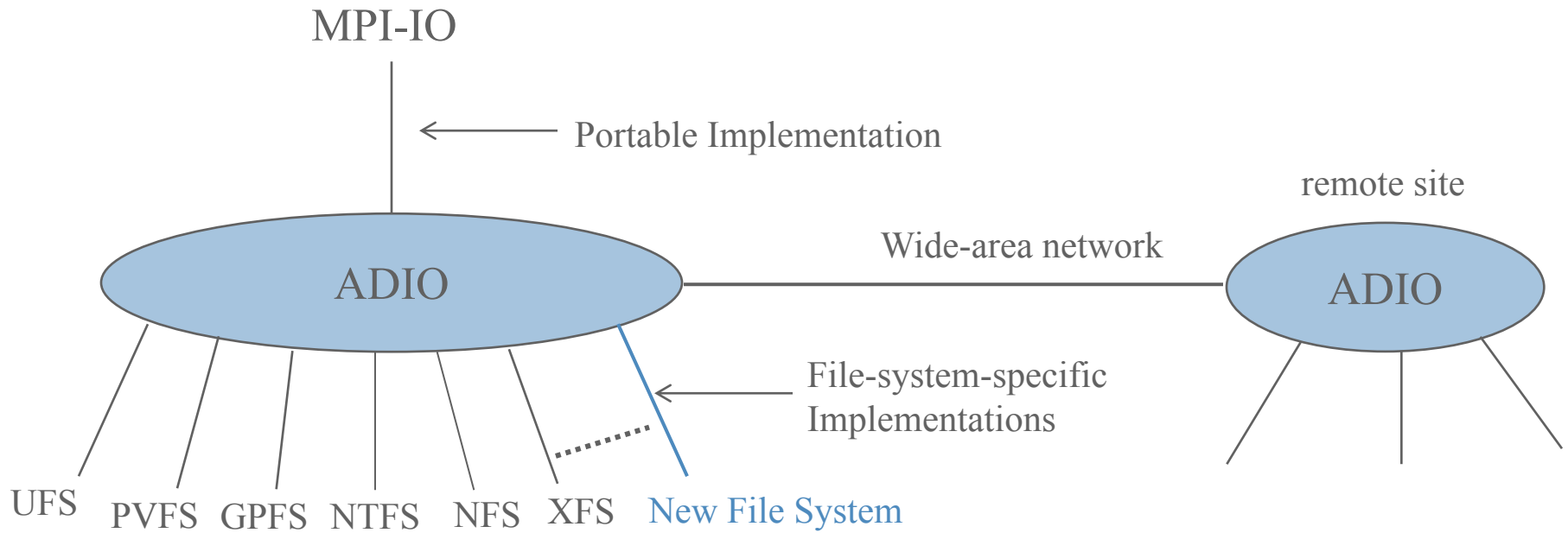
The Origins (4)

- In the meanwhile, the MPI Forum had resumed meeting to define new features that would be part of MPI-2
- The MPI-IO Committee eventually merged with the MPI Forum, and, from the summer of 1996, the MPI-IO activities took place in the context of the MPI Forum meetings
- The MPI Forum used the latest version of the existing MPI-IO specification as a starting point for the I/O chapter in MPI-2
- The I/O chapter evolved over many meetings of the Forum and was released along with the rest of MPI-2 in July 1997

The Origins (5)

- Bill Nitzberg was the chair of the I/O chapter in MPI-2. He navigated the MPI Forum process very skillfully.
- I joined Argonne as a postdoc in May 1995, working with Bill Gropp and Rusty Lusk, and participated in the MPI-IO definition in the MPI-2 Forum
- I also developed the ROMIO implementation of MPI-IO, which tracked the evolving MPI-IO spec (similar to the way MPICH tracked the MPI spec)

ROMIO Implementation of MPI-IO

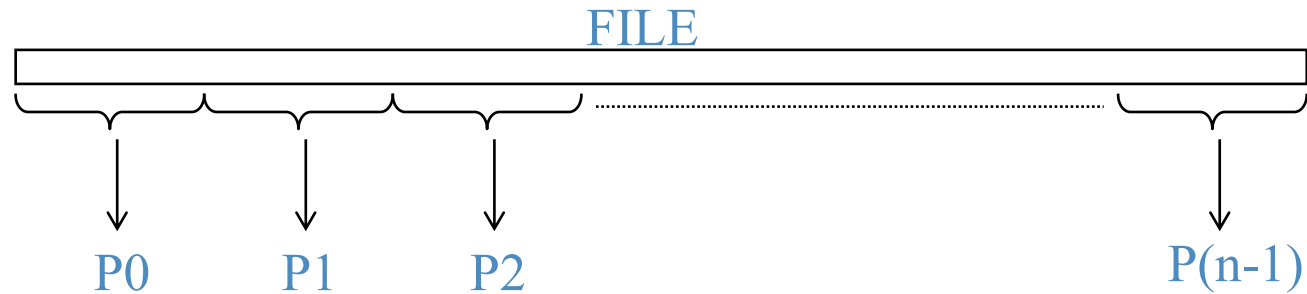


ROMIO is available as part of all MPI implementations

Basic MPI-IO

- MPI-IO provided a familiar file I/O interface (open, close, read, write, seek) and some additional features
- To MPI users, it was like MPI
- To I/O developers, it was a file I/O interface with an MPI flavor

Using MPI-IO




```
MPI_File fh;
MPI_Status status;

MPI_Comm_rank(MPI_COMM_WORLD, &rank);
MPI_Comm_size(MPI_COMM_WORLD, &nprocs);

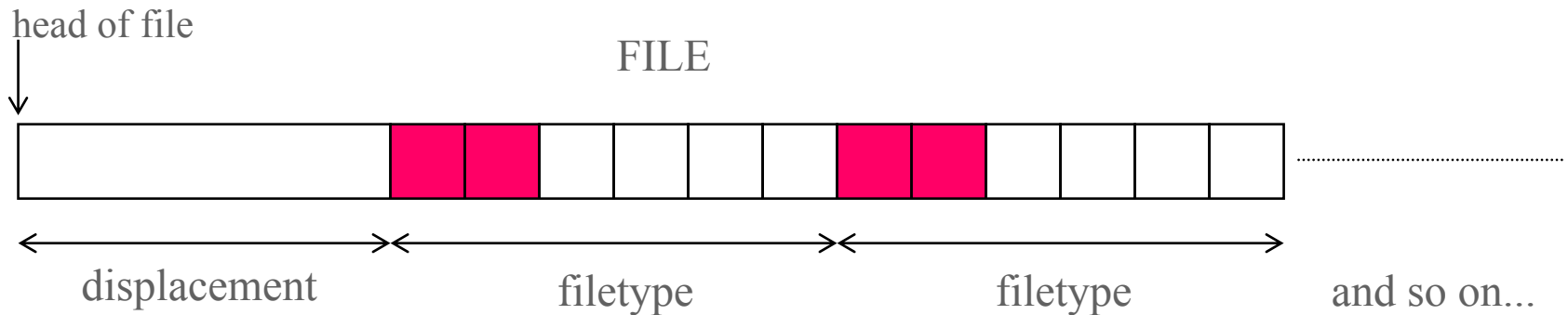
bufsize = FILESIZE/nprocs;
nints = bufsize/sizeof(int);

MPI_File_open(MPI_COMM_WORLD, "/pfs/datafile",
              MPI_MODE_RDONLY, MPI_INFO_NULL, &fh);
MPI_File_seek(fh, rank * bufsize, MPI_SEEK_SET);
MPI_File_read(fh, buf, nints, MPI_INT, &status);
MPI_File_close(&fh);
```

New Feature: File views for noncontiguous data

 etype = **MPI_INT**

 filetype = two **MPI_INT**s followed by
a gap of four **MPI_INT**s



What turned out to be useful or widely used

- Collective reads/writes (MPI_File_read_all / write_all)
- Nonblocking reads/writes (MPI_File_iread / iwrite)
- Explicit offset reads/writes (MPI_File_read_at / write_at) and their collective and nonblocking versions
- MPI_Info hints
- Well defined set of consistency semantics

Features that are not widely used

- File interoperability
 - Internal, external32 data representations
 - User-defined data representations
 - Not used because higher-level libraries that use MPI-IO have their own file format (HDF, PnetCDF)
- Shared file pointers
- Split collective I/O (`MPI_File_read_all_begin / end`)
 - Restricted form of nonblocking collective I/O
 - MPI 3.1 has added regular nonblocking collective I/O functions (`MPI_File_iread_all`, etc.)

Current Status

- MPI-IO is still widely used by applications for parallel I/O, either directly or indirectly through high-level I/O libraries (such as HDF-5, PnetCDF, ADIOS) that use MPI-IO
- Many file systems are supported (GPFS, Lustre, ...)
- Others contribute code for new file systems
- Many papers have been written (by many people) on optimizations that can be done within an MPI-IO implementation

Future

- Work is always needed in tuning an implementation for the specific I/O configurations of large supercomputing systems
- Giving users too many options (in the form of hints they can pass) is not useful – they don't use them
 - The implementation must figure out as many of the best values it can
- The presence of burst buffers (NVRAM) on a system changes the economics of small v/s large I/O operations
 - There is less need to aggregate small I/O operations into large ones, or the cutoff point is different
 - The implementation needs more smarts to know what is the best approach

Summary

- Twenty years have flown by since the MPI-2 days
- It has been my privilege to be involved in many aspects of MPI (in addition to MPI-IO)
 - MPICH implementation
 - MPI-2.1, 2.2, and 3.0 standardization
 - RMA and collectives
- Thanks to many colleagues and friends whom I have worked with over the years