Implementing MPI: the 1994 MPI Implementors' Workshop

William Gropp and Ewing Lusk *
Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439

Abstract

In September of 1994 a workshop was held at Argonne National Laboratory on implementation issues for MPI. MPI is a standard message-passing library interface developed during 1993 and 1994 by the MPI Forum, a broadly based group of parallel computing vendors, parallel library writers, and application scientists. The purpose of the Workshop was to gather together those vendors and others actively engaged in implementing the Standard in order to discuss issues raised by the implementation process. A secondary aim was to explore the possibility of pooling some efforts in order to speed and ease the adoption of MPI. Here we briefly describe the implementation efforts that were presented at the Workshop, summarize the discussions that took place on the Standard itself and implementation issues, and offer some general conclusions about the state of the MPI implementation effort worldwide.

1 Introduction

During 1993 and 1994, the MPI Forum, a group of parallel computer vendors, library writers, and applications scientists, developed a specification for a standard message-passing library interface. The goal was to agree on an informal portable standard that would be attractive to both implementors and users, address some of the shortcomings of existing message-passing library specifications, and promote the development of parallel software libraries. The MPI Standard [4, 5, 12], was finalized in May of 1994, and multiple implementation projects have begun.

In September of 1994, we organized an MPI Implementors' Workshop at Argonne National Laboratory. It seemed like an appropriate time, since a number of implementations were well under way, yet few were completed. Topics that were anticipated to be of particular interest to the MPI implementation community included:

- joint assessment of the status of MPI implementation worldwide
- the Standard document itself, and how to deal consistently with the published errors and problems that implementation efforts had brought out
- implementation techniques, particularly advanced or unusual ones
- extensions to MPI, with the intention of maintaining consistency even for functionality that is not part of the standard
- interoperability among different implementations
- the possibility of sharing test code, both to spread the labor of developing it and further enhancing the portability of MPI applications.

All of these discussions occurred at the Workshop, as we describe below. In the following sections we will list the implementation efforts represented at the Workshop, summarize some of the presentations and discussions, and draw some conclusions about where MPI stands today.

2 Participants

An enjoyable benefit of organizing the Workshop was the discovery of a number of significant MPI implementation efforts that we had previously been unaware of. At the time of the workshop, the MPI implementations shown in Table 1 were all in progress. Others may exist as well.

Other interested hardware and software companies who were represented at the Workshop were

*This work was supported by the Office of Scientific Computing, U.S. Department of Energy, under Contract W-31-109-Eng-38.

Vendor Implementations
IBM Research (MPI-F)
IBM Kingston
Intel SSD
Cray Research
Meiko, Inc.
Kendall Square Research
NEC
Convex
Hughes Aircraft
Portable Implementations
Argonne–Mississippi State (MPICH)
Ohio supercomputer Center (LAM)
University of Edinburgh
Technical University of Munich
University of Illinois

Table 1: Current MPI implementations

Sun, Hewlett-Packard, Myricom (makers of high-performance network switches) and PALLAS (a German software company). Sandia National Laboratory plans an implementation for their Intel Paragon running SUNMOS.

The vendor implementations are not yet supported products of their respective companies. The portable implementations are, or will be, freely available.

3 Workshop Content

The Workshop itself consisted of intermixed presentations and discussion sessions. We summarize here some of the presentations on specific implementations and other topics, and follow them with summaries of some of the discussions that took place.

3.1 Presentations

Bill Gropp, Argonne National Laboratory. This talk described the Argonne–Mississippi State MPI implementation (MPICH) [9] and some of the lessons learned from the implementation process. This implementation obtains portability without giving up efficiency by the use of an Abstract Device Interface (ADI) that renders most of the implementation portable and thus usable by others [10]. It currently runs directly on several parallel supercomputers (IBM, Intel, TMC) and via `p4` [3] and PVM [7] on other parallel machines and workstation networks.

Greg Burns, Ohio Supercomputer Center. LAM is an implementation of MPI for workstations layered on the Trollius distributed system. The talk stimulated discussion of the "progress" rules in the Standard, and whether they could be implemented on systems without asynchronous I/O operations and interrupts or threads. More information on LAM can be found in [2].

Gordon Smith, Edinburgh Parallel Computing Centre. The EPCC is well known for its CHIMP portability library [1]. It is currently involved in two MPI implementations, one on top of CHIMP, which uses parts of MPICH, and the other a joint project with Cray with the goal of producing a native implementation on the T3D, using the shared-memory `put` and `get` operations. EPCC is interested in collaborations on user-level documentation, to aid in converting their users to MPI.

Paul Pierce, Intel Scientific Supercomputer Division. Intel has demonstrated good performance of an experimental MPI implementation that is very close to that of NX itself. In the long run, Intel is interested in using MPI as the user-level interface to very lightweight communication protocols.

James Cownie, Meiko, Inc. The Meiko CS-2 has a communications co-processor (the Elan) that can be used to advantage in the implementation of many of MPI's operations. Cownie discussed the implementation of MPI's contexts with "tports" on the Elan that would create a separate end point for communication in each context. In this way, each library (using a separate context) would see no performance impact when a new library is added. Cownie pointed out that the "persistent handles" in MPI do not really offer the expected performance advantages on hardware that can provide remote-memory `put/get` operations.

Koichi Konishi, NEC. NEC in Japan is currently pursuing two different implementations. One of them is only a partial implementation, on a proprietary system, and the other is a full MPI implemented on top of MACH, using a memory-mapped network device. His talk stimulated discussion about what happens if an `MPI_Isend` has no corresponding `MPI_Wait`, and similar situations. He remarked that his users were glad to see inclusion of `MPI_Bsend`, for explicit buffering control.

Lloyd Lewins, Hughes Aerospace and Electronics Co. This talk described an unusual implementation of MPI, on a distributed-shared-memory processor for embedded MPI applications. The goal is a real-time MPI execution environment. Hughes is using the ANL-MSU implementation and modifying the ADI to suit their environment. Some of the changes are general improvements and will be incorporated in the the next release of MPICH.

Ashish Singhai, University of Illinois. A very new effort is being made to provide an application interface to ATM networks, at the level of AAL5. MPI has been selected as the message-passing interface and work will begin soon.

Hans-Christian Hoppe, PALLAS. PALLAS is a software company in Berlin that has a number of commercial customers. It is not only interested in marketing a commercial portable version of MPI but in providing user training courses and tools for converting programs from other message-passing libraries to MPI.

Steve Breit, Kendall Square Research. KSR is starting with MPICH as the beginning of their implementation, but requires better support for collective operations. KSR's memory architecture makes a naive implementation easy, but KSR is concerned about very precise buffer management for peak performance.

Hubertus Franke, IBM Research (Yorktown Heights). IBM was one of the first companies with a serious MPI implementation, which initially borrowed code from MPICH. The implementation is complete, and achieves performance equal to or better than the IBM proprietary library MPL on the SP1 and SP2 [6]. The talk provided significant levels of detail about the methods used to achieve this performance.

Bill Tuel, IBM Kingston. IBM is planning to offer MPI on the SP2 with the same level of tool support (debugging, performance analysis) as for its proprietary MPL library. IBM is interested in sharing both thoughts and effort on testing and validation of MPI implementations. Future extensions of interest include parallel I/O, dynamic task creation, and heterogeneity.

Peter Rigsbee, Cray Research. Cray is supporting the development of an MPI implementation by the Edinburgh Parallel Computing Centre for the Cray T3D.

Nan Boden, Myrinet. Myrinet is a new company that makes high-performance switches. Among other things, these switches can be used to assemble parallel supercomputers out of workstations. Myrinet is seeking a programming interface for such machines.

David Culler, University of California (Berkeley). This talk presented a careful analysis of the MPICH Abstract Device Interface, with particular attention to the gains that could result from an active-message approach on machines that supported this lightweight protocol or similar ones. Few changes actually need to be made to the ADI, and its developers will be working with Culler to explore this approach to MPI implementation.

Tony Skjellum, Mississippi State University. Mississippi State is a partner with Argonne in development of the MPICH implementation, but is also independently developing extensions and tools. One particular effort is in the merging of the MPI and PVM programming interfaces. In the area of extensions, a library is being developed that will define and implement a set of collective operations for intercommunicators.

3.2 Discussions

The MPI Standard document. The final version of the MPI Standard has a date of May 5, 1994. It is available from a variety of sources [4, 8, 12]. There is an official "Errata sheet" [13, 14], pointing out a small set of errors in the Standard and raising a number of issues for further discussion. There was general agreement on the existing Errata. There was also agreement on an "errata erratum": the `max`, `min`, `maxloc`, and `minloc` collective operations for the complex and double-complex datatypes are to be dropped altogether from MPI, not just modified as described in the Errata. The group agreed with the Errata that `MPI_NULL_COPY_FN` and `MPI_NULL_DELETE_FN` should be separate, rather than combined into `MPI_NULL_FN` as they are now. Discussion of how to handle `MPI_BOTTOM` in Fortran did not reach a conclusion.

There was general agreement that except for minor matters like this, the Standard document was an adequate guide for implementors to work from.

Dynamic process creation and related topics. This discussion centered on two proposals for what an MPI_Spawn function, for dynamic process creation, might look like, and requirements for its behavior. This needs to be a collective operation (in some communicator) to ensure safety. Most of the discussion was about whether the new communicator should be an inter- or and intra-communicator and whether it should be possible to dynamically alter the original MPI_COMM_WORLD. Since MPI_Intercomm_merge is part of the Standard, the various proposals all offer the same functionality, but differ in how convenient they are for various applications. There was also discussion about whether the exiting of a spawned process needed to be a collective operation.

There was a brief discussion of interrupt-driven receives (`hrecv` on Intel, `recvncall` on IBM). The consensus was that such an extension to MPI is not needed because most systems will have thread packages that interact properly with their message-passing libraries in the near future.

It was pointed out that the presence of MPI_Intercomm_merge means that communicators cannot in general be assumed to contain only groups of processes that are "uniform" in some way, such as running on the same type of processor or being connected in the same way.

Sharable MPI Resources. What are the existing sources of information about MPI? The following is a partial list of source material. some of these, particularly WWW ages, include pointers to other resources.

- The Standard itself:
  - As a Technical report: U. of T. report [4]
  - As postscript for ftp: at `info.mcs.anl.gov` in `pub/mpi/mpi-report.ps`.
  - As hypertext on the World Wide Web: `http://www.mcs.anl.gov/mpi`
  - As a journal article: in the Fall issue of the Journal of Supercomputing Applications [12]
- MPI Forum discussions
  - The MPI Forum email discussions and both current and earlier versions of the Standard are available from `netlib`.
- Books:
  - *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, by Gropp, Lusk, and Skjellum [11].
  - MPI Annotated Reference Manual, by Otto, et al., in preparation.
- Newsgroup:
  - `comp.parallel.mpi`
- Mailing lists:
  - `mpi-comm@cs.utk.edu`: the MPI Forum discussion list.
  - `mpi-impl@mcs.anl.gov`: the implementors' discussion list.
- Implementations available by `ftp`:
  - MPICH is available by anonymous `ftp` from `info.mcs.anl.gov` in the directory `pub/mpi/mpich`, file `mpich.*.tar.Z`.
  - LAM is available by anonymous `ftp` from `bag.osc.edu` in the directory `pub/lam`.
  - The CHIMP version of MPI is available by anonymous `ftp` from `ftp.epcc.ed.ac.uk` in the directory `pub/chimp/release`.
- Test code repository (new):
  - `ftp//:info.mcs.anl.gov/pub/mpi-test`

Continuing collaboration. This discussion was about how to continue the informal collaboration among MPI implementors that was established at this workshop. For example, collaboration is needed on an "external MPI implementation interface" to allow interoperability among different MPI implementations. This would require agreement on at least three levels: the envelope format and protocol, the data format translation necessary to support heterogeneity (XDR replacement), and the underlying transmission protocol (such as TCP or UDP).

There was universal agreement that it would be useful to form a common test suite for the purpose of ensuring portability of MPI applications and to share the cost of producing such a test suite. There was tentative agreement to set this up at Argonne, in `ftp//:info.mcs.anl.gov/pub/mpi-test`.

4 Conclusions

When the MPI Forum was creating the specification for MPI, there was some concern about the level of interest on the part of vendors and other implementors in providing robust versions of it so that application programmers and library writers could reap the benefits. One obvious conclusion from this workshop is that both vendors and others have found MPI to be important and useful enough, and enough of an advance over existing message-passing systems, to invest substantial effort in producing high-quality implementations. The number and variety of MPI implementations represented at the workshop attest to the enthusiasm with which MPI is being adopted, not only in the U.S, but also in Europe and Asia.

The workshop discussions revealed that the Standard document itself, though not perfect, is an adequate guide to implementation. Work on complete implementations is proceeding quickly, putting to rest any concerns that the MPI specification is too large or difficult to implement. This is no coincidence since many implementors participated in the MPI Forum itself.

It now seems clear that within a year there will a number of vendor implementations of MPI to choose from, fully supported by tools. In the meantime, complete, public implementations offer the opportunity for both library writers and applications scientists to begin using MPI now on many different parallel computing environments. The formation of a community, which will continue to collaborate, of those specializing in MPI implementation bodes well for the stability of the MPI Standard.

References

[1] The CHIMP release. World Wide Web. `ftp://ftp.epcc.ed.ac.uk/pub/chimp/release`.

[2] Collected LAM documents. World Wide Web. `ftp://tbag.osc.edu/pub/lam`.

[3] Ralph Butler and Ewing Lusk. Monitors, messages, and clusters: The p4 parallel programming system. *Parallel Computing*, 20:547–564, April 1994. (Also Argonne National Laboratory Mathematics and Computer Science Division preprint P362-0493).

[4] Message Passing Interface Forum. MPI: A message-passing interface standard. Computer Science Dept. Technical Report CS-94-230, University of Tennessee, Knoxville, TN, 1994.

[5] The MPI Forum. MPI: A message-passing interface standard. World Wide Web. `http://www.mcs.anl.gov/mpi/mpi-report/mpi-report.html`.

[6] Hubertus Franke, Peter Hochschild, Pratap Pattnaik, Jean-Pierre Prost, and Marc Snir. MPI-F: An MPI prototype implementation on IBM-SP1. (contact: Hubertus Franke, frankeh@watson.ibm.com), 1994.

[7] Al Geist, Adam Beguelin, Jack Dongarra, Weicheng Jiang, Bob Manchek, and Vaidy Sunderam. *PVM: Parallel Virtual Machine—A User's Guide and Tutorial for Network Parallel Computing*. MIT Press, 1994.

[8] William Gropp and Ewing Lusk. Message passing interface. World Wide Web. `http://www.mcs.anl.gov/mpi`.

[9] William Gropp and Ewing Lusk. MPICH. World Wide Web. `ftp://info.mcs.anl.gov/pub/mpi`.

[10] William Gropp and Ewing Lusk. An abstract device definition to support the implementation of a high-level message-passing interface. Technical Report MCS-P342-1193, Argonne National Laboratory, 1993.

[11] William Gropp, Ewing Lusk, and Anthony Skjellum. *Using MPI: Portable Parallel Programming with the Message Passing Interface*. MIT Press, 1994.

[12] Message Passing Interface Forum. MPI: A message-passing interface standard. *International Journal of Supercomputer Applications*, 8(3/4), 1994.

[13] Steve Otto. Errata sheet for MPI specification. World Wide Web. `http://www.mcs.anl.gov/mpi/errata.ps`.

[14] Steve Otto. Errata sheet for MPI specification. available by `ftp`, July 1994.