Reviews

Using MPI—Portable Parallel Programming with the Message-Passing Interface, by William Gropp, Ewing Lusk, & Anthony Skjellum. ISBN 0-262-57104-8, 1994, $24.95, 307 pp., softbound. Available from The MIT Press, Cambridge, MA.

One of the major barriers that has kept scientific programmers away from parallel computers is the lack of a standard programming paradigm. MPI, the message-passing interface standard, was developed specifically to provide a standard set of message-passing calls that enable programmers to write message-passing applications that are portable and maintainable (e.g., existing MPI calls will not change as has been the case for other libraries such as PVM [1] and [2]). A problem with the MPI standard, however, has been the lack of documentation, other than the standard document itself. The first major book to address this need is *Using MPI—Portable Parallel Programming with the Message-Passing Interface*, by Gropp, Lusk, and Skjellum. The first three chapters of this book provide a good starting point for MPI users looking for a guide with a more tutorial approach than the MPI standard document. Specifically, the "six-function version of MPI" described in Chapter 2 will be invaluable to programmers, and is very hard to derive from the standard document (the six functions come out of at least three chapters in the standard). Unfortunately, the rest of *Using MPI* is schizophrenic in its approach, and it is unclear whether it is targeted at beginners, more advanced users, or implementors of MPI. In addition, because of its structure, it is unlikely that *Using MPI* will be useful as a reference book once you are comfortable with the basic concepts in MPI.

Using MPI starts out with a simple subset of MPI and proceeds to more complex examples. However, since this book only lightly covers the basics of message passing, and quickly relates them to the specifics in MPI, even these beginning chapters are best suited for someone who has had previous experience writing message-passing programs. Examples are all simple scientific kernels in C or Fortran 77. Therefore, a knowledge of C or Fortran is required for this book (as it is for MPI), and knowing both is helpful. The examples in Chapter 3 demonstrate the basic functionality of MPI, without getting bogged down in more advanced features. Chapters 4 and 5 cover more advanced programming topics that may not initially be of interest to many parallel programmers, but may be helpful down the line when you are more comfortable with MPI. Examples include information about timing and performance monitoring. In addition, some examples use tools provided with the authors' implementation of MPI (MPICH, available via FTP at info.mcs.anl.gov) that are not available in all MPI libraries. These tools include graphics functions from the MPE library (a library of MPI extensions that is part of MPICH) as well as the Upshot performance monitoring tool. This reliance on MPICH simplifies some of the ambiguities in the MPI standard. In particular, since MPI only standardizes message passing, there is no standardization of program loading (i.e., what command to use to start your parallel job) or I/O. Unfortunately, these features can be different for each implementation of MPI, so there is no perfect way to address them in a general MPI book. This means that supplemental documentation will be necessary when using a different MPI implementation. In addition, some of the tools demonstrated in the book do not exist in every MPI implementation. All of the examples contained in the book are available via anonymous FTP.

After Chapter 5, the book diverts from its focus on scientific application programmers and pre-

Reviewed November 1995

Scientific Programming, Vol. 5, pp. 275–276 (1996)
CCC 1058-9244/96/030275-02

sents some specific topics that will not be of interest to everyone. Chapter 6 covers the library support in MPI, and gives an example of a simple scientific library. Chapter 7 covers some of the other advanced features present in MPI, in particular intercommunicators, advanced collective operations, heterogeneous computing, error handling, environmental inquiry, and profiling. This chapter will be very useful to a certain part of the application programmer community that needs these functions, but it can be skipped by most programmers. Chapter 8 discusses issues about implementing MPI, and should really be relegated to either a separate paper or an appendix. However, it might still be useful for providing programmers insight into the performance of MPI implementations. Chapter 9 presents a guide for porting from an existing message-passing library to MPI. Porting guides are presented for the following libraries: Intel NX, IBM EUI (also known as MPL), Thinking Machine's CMMD, Express, PVM 2.4.x, PVM 3.2.x, p4, PARMACS, TCMSG, Chameleon, and Zipcode. The final chapter covers some future issues, many of which are being addressed in the MPI-2 effort which has already started. Finally, the appendices provide reference for MPI function calls and pointers to MPI resources on the Internet.

Probably the biggest deficiency of *Using MPI* is that it tries to cover too much, instead of concentrating on simply being a user's guide for application programmers. However, even with these problems, it should be useful to anyone who wants to use MPI, as long as you don't expect to read the entire book cover to cover (i.e., you should concentrate on the chapters specific to what you need to know). If you are new to MPI, I would definitely recommend that you read the first three chapters of the book. In addition, if you are familiar with other message-passing libraries, you should look at the appropriate sections of Chapter 9. Beyond that, I would recommend that you only look at the advanced features of MPI if you think you will need them. *Using MPI* is much better as an MPI user's guide than the standard document itself, but it still misses the mark.

REFERENCES

[1] V. S. Sunderam. "PVM: A framework for parallel distributed computing," *Concurrency Practice Exp.*, vol. 2, pp. 315–339, Dec. 1990.

[2] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, and V. S. Sunderam, *PVM: Parallel Virtual Machine: A Users' Guide and Tutorial for Network Parallel Computing, Scientific and Engineering Computation Series*. Cambridge, MA: MIT Press, 1994.

Samuel A. Fineberg
MRJ, Inc.
NASA Ames Research Center M/S 258-6
Moffett Field, CA 94035-1000

e-mail:fineberg@nas.nasa.gov
Phone: (415)604-4319
Fax: (415)966-8669