Standards for Message-Passing in a Distributed Memory Environment

David W. Walker

CRPC-TR92230
August 1992

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

ORNL/TM-12147

Engineering Physics and Mathematics Division

Mathematical Sciences Section

STANDARDS FOR MESSAGE-PASSING IN A DISTRIBUTED MEMORY ENVIRONMENT

David W. Walker

Mathematical Sciences Section
Oak Ridge National Laboratory
P.O. Box 2008, Bldg. 6012
Oak Ridge, TN 37831-6367

Date Published: August 1992

Research was supported by the Applied Mathematical Sciences Research Program of the Office of Energy Research, U.S. Department of Energy.

Prepared by the
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831
managed by
Martin Marietta Energy Systems, Inc.
for the
U.S. DEPARTMENT OF ENERGY
under Contract No. DE-AC05-84OR21400

Contents

STANDARDS FOR MESSAGE-PASSING IN A DISTRIBUTED MEMORY ENVIRONMENT

David W. Walker

Abstract

This report presents a summary of the main ideas presented at the First CRPC Workshop on Standards for Message Passing in a Distributed Memory Environment, held April 29-30, 1992, in Williamsburg, Virginia. This workshop attracted 68 attendees including representatives from major hardware and software vendors, and was the first in a series of workshops sponsored by the Center for Research on Parallel Computation. The aim of this series of workshops is to develop and implement a standard for message passing on distributed memory concurrent computers, thereby making it easier to develop efficient, portable application codes for such machines. The report discusses the main issues raised in the CRPC workshop, and describes proposed desirable features of a message passing standard for distributed memory environments.

1. Introduction

This report gives an overview of the main ideas presented at the First CRPC Workshop on Standards for Message Passing in a Distributed Memory Environment, held April 29–30, 1992, at the Hilton Conference Center in Williamsburg, Virginia. The workshop, which was generously sponsored by the Center for Research on Parallel Computing (CRPC), was attended by a total of 68 invited participants from universities, government laboratories, and hardware and software vendors. The aim of the workshop was to assess the need for a message-passing standard on distributed memory computing systems, and to establish a process for defining and implementing the standard. In addition, the workshop discussed the important components that should be included in such a standard. The workshop included 19 talks divided among 5 sessions, and a panel discussion session. It is not the purpose of this report to summarize each of the talks individually, but rather to present the main ideas that arose from the talks, and the subsequent discussion. The workshop program, and a list of attendees, are given in Appendices A and B, respectively.

Among the general matters discussed was the necessity of defining a global standard, rather than just a U.S. standard. The importance of interacting with ongoing standardization efforts in Europe was stressed. This ongoing work was described in the first of two talks by Rolf Hempel of GMD, who discussed the role played by the European Community in fostering parallel computing standards through its ESPRIT research program. It was also generally agreed that vendors should be closely involved in the standardization effort, in order to ensure that whatever message-passing standard emerges can and will be implemented efficiently on commercial distributed memory computing systems.

2. The Need for a Standard

An important issue addressed near the start of the workshop was whether a message-passing standard is necessary. It could be argued that the most difficult and time-consuming aspects of implementing an application on a distributed memory computing system are

1. devising a correct parallel program, and

2. optimizing the code to get efficient and scalable performance.

Thus, the argument goes, in porting a code between two distributed memory computing systems the time spent in replacing the message-passing calls of one system with those of the other is negligible, and hence a standard doesn't gain you much. From this viewpoint issues such as algorithmic correctness, the need for tools to aid in the optimization of parallel programs, and the development of distributed memory computer hardware with low communication costs, are the most important issues facing the research community. In defining a message-passing standard now, we anticipate advances in these areas that will make the imposition of the standard at a later date useful and worthwhile. Of course, the main objectives of a message-passing standard are portability and ease-of-use. It was also pointed out at the workshop that, by providing high-level routines and/or abstractions, a message-passing standard can reduce the likelihood of programming errors, thereby enhancing program correctness. Another point made was that the definition of a message-passing standard would provide vendors with a clearly defined set of routines that they could implement efficiently at a low level, or even provide hardware support for, in some cases. Thus, a message-passing standard not only provides portability and ease-of-use, but also addresses to a limited extent the issues of program correctness and performance.

There was some concern expressed that standards not be imposed too early, i.e., while the desired functionality is still uncertain. Clearly there is little point in having a "standard" that must be modified on a short timescale. It emerged during the workshop that there is a large measure of agreement over what should be included in a message-passing standard. Thus, the prevailing opinion was that a standard is needed, and that now is a good time to begin the process of defining it.

3. Features of the Standard

It is possible to consider defining a message-passing standard at a number of levels. At the lowest level, closest to the hardware, might be syntactically simple routines for moving packets along wires. Above this channel-addressed level might be a process-addressed level (where a "process" may, or may not, be equivalent to a "processor"), such as that defined by NX or Vertex on the IPSC and nCUBE machines, the commercially-available *Express* communication environment, or the PARMACS message-passing macros that form the basis of a draft standard for message-passing in Europe. Higher-level abstractions, for example, Linda, MetaMP, or Shared Objects, would lie above this level. Each level could be built using the level beneath, provided that the overhead in doing this was sufficiently low that the cumulative overhead incurred at the higher levels was small. These successive software levels form a series of layers, that with some stretch of the imagination resemble the multiple skins of an onion, with the hardware being at the center. We, therefore, call this the "Onion Skin Model" of the distributed communication environment. One of the issues discussed at the workshop was at what level is it best to try to impose a standard. It was noted that different people might favor different standards. For example, a non-expert user would prefer to use high-level abstractions, such as virtual shared memory, so that details of the message-passing are hidden. An expert application developer might be prepared to sacrifice some ease-of-use for additional speed, and so would prefer a standard that provides a set of efficient primitives for point-to-point message-passing, together with some global operations. Finally, a compiler writer would like to produce a portable parallel compiler, and would like to use small, fast messages such as might be provided by a low-level standard.

If the Onion Skin model is valid, then it makes sense to impose a standard that is also layered. However, it was pointed out that the hardware of different distributed memory computing systems is sufficiently varied that it is difficult to impose a low-level standard that is efficient on all machines. Therefore, it is more appropriate to define a standard at an intermediate level, and to implement this as efficiently as possible on each machine. There is still the possibility of defining higher-level standards on top of this intermediate level. Thus, the intermediate-level standard will be open and extendable.

Many of the talks at the workshop focused on an intermediate-level standard based on point-to-point message passing, together with some higher-level, collective communication routines. The general consensus that emerged was that the following were desirable features of a message-passing standard,

- Point-to-point message passing between processes (or processors) with:
  - message selectivity by type and source
  - message contexts
  - blocking and nonblocking communication primitives
  - support for communication of non-contiguous data
- Ability to define process groups
- Global reduction operations
- Gather, scatter, and scatter-with-add routines
- Collective communication primitives such as shift, broadcast, and concatenate
- Support for heterogeneous distributed computing systems

Some of these features require further elucidation.

3.1. Message Contexts

Often a parallel program divides naturally into different computational phases. Message contexts can be used to prevent nonblocking messages from different phases interfering with one another without the need for a time-consuming barrier synchronization between phases.

3.2. Blocking and Nonblocking Communication

The receipt of a message is said to be blocking if the receiving process suspends execution until all of the message has been received. A nonblocking receive takes place in two phases. In the first a receive is posted on the receiving process, that is, the user provides a buffer that is to be used to store a specified incoming message. The receiving process can then continue to do useful work while waiting for the message to arrive. However, before the data in the incoming message can be used the receiving process must suspend execution until the message has arrived and been placed in the buffer supplied by the user. This is the second phase of a nonblocking receive. A blocking receive is conceptually the same as a nonblocking receive in which no useful work is done between the two phases.

The above method of using nonblocking receives is commonly used when the maximum amount of work that could be done between posting the receive and actually using the received data is know at compile time. In more dynamic situations there may be an almost arbitrary amount of work that a process could do until an anticipated message arrives. In such cases it is common to periodically check whether the message has arrived by calling a low overhead probe routine. As long as the probe routine indicates that the message has not arrived the process continues to do useful work, but once the message arrives it is processed.

The sending of a message is said to be blocking if the sending process suspends execution until all of the message has been received. There are (at least) two types of nonblocking send. In one type the sending process suspends execution until it is safe to overwrite the message buffer, i.e., until the buffer is guaranteed to be non-volatile. We can call this a partially blocking send. A fully nonblocking send takes place in two phases. In the first phase the user supplies a message buffer on the sending process and transmission of this buffer to the receiving process is initiated. While the message is in transit the sending process can continue to do useful work, but during this time the message buffer is volatile, and it is a programming error to change it in any way. In the second phase of a nonblocking send the sending process suspends execution until the message buffer is no longer volatile. A partially blocking send is conceptually the same as a nonblocking send in which no useful work is done between the two phases.

In point-to-point communication between two processes any combination of communication modes can be used on the receiving and sending processes. Fully blocking communication is often referred to as "synchronous" communication.

3.3. Noncontiguous Messages

Two methods for sending noncontiguous data from one process to another in a single message were described at the workshop. In the first method the message to be sent is made up of blocks of data separated by a fixed stride in the memory of the sending process. On the receiving process the message is received into a user-supplied buffer in blocks of data separated by a fixed stride in memory. In general, the block size and stride do not have to be the same on the receiving and sending processes. This type of communication could be used, for example, to communicate a row of a distributed matrix that is stored by columns. In the second method the outgoing message on the sending process is specified by a vector, each element of which is a structure consisting of a pointer and an integer. The message is composed by looking at the first structure in the vector, and, starting at the memory location given by the pointer, copying the number of bytes specified by the corresponding integer into the message buffer. Next the data specified by the second structure in the vector is added to the message buffer directly after that of the first, and so on for all structures in the vector. On the receiving process the incoming message can be unpacked into user memory using a similar vector of structures. This type of communication could be used in certain types of gather/scatter operations in which the distributed object from which data are being gathered and/or to which data are being scattered has a regular decomposition, for example, the Cartesian grid typically used in particle-in-cell simulations. Clearly, the first method using a constant stride is a special case of the second method.

3.4. Process Subgroups

In some applications it is advantageous to be able to dynamically partition the processes into process subgroups that may, or may not, overlap. This permits functional parallelism to be exploited, by allowing different groups of processes to work on different subtasks in an application.

3.5. Reduction Operations

Given a set of vectors with the same data distribution a reduction operation combines the elements of each vector in a pairwise fashion using an associative, commutative reduction function, and distributes the result to all processes. Thus, given the N elements of vector V, and a reduction function, $\oplus$, the result of the reduction operation would be,

$$A = V_1 \oplus V_2 \oplus \cdots \oplus V_N \tag{1}$$

3.6. Gather/Scatter Routines

Given distributed vectors X and A of length N, and an indirection vector, K, of integers, the gather, scatter, and scatter-with-add are most simply typified as follows:

$$\begin{array}{rclr} X(I) & = & A(K(I)) & \text{GATHER} \\ A(K(I)) & = & X(I) & \text{SCATTER} \\ A(K(I)) & = & A(K(I)) + X(I) & \text{SCATTER-WITH-ADD} \end{array} \qquad (2)$$

for $I = 1, \ldots, N$. This is readily extended to the case of multidimensional arrays.

A gather operation executed loosely synchronously on all processes would examine the indirection array, K, on each process and gather to each process those elements of the array indexed by its indirection array. Clearly, such a gather operation would need to know how the array is distributed over the processes. This type of gather operation differs from that described in Sec. 3.3, which is really a coordinated gather/scatter operation between two specific processes.

A scatter operation can be defined in a similar way, except in this case the indirection array on each processor indicates to which array elements data are to be scattered. For consistency no two entries in the indirection arrays of all processes may refer to the same target array element. Thus this type of scatter operation can be used to permute an array.

The scatter-with-add operation is similar to the scatter operation except that the restriction on the uniqueness of target array elements pointed to by the indirection arrays is relaxed, and data scattered to the same array element are additively accumulated.

3.7. Collective Communication

Collective communication routines involve the coordinated exchange of data between processes in a predictable, regular way. Examples include shifting an array along a specified array axis, replicating an array along a specified array axis, one-to-all broadcasts, and all-to-all broadcasts (or concatenation).

3.8. Support for Heterogeneous Computing

In the context of a message-passing standard, support for heterogeneous computing means that it should be possible for the user to communicate data transparently between processes residing on different types of processor, without having to worry about the processors having different ways of internally representing the data. In a broader context it is desirable to define a standard for heterogeneous computing, but it should be noted that this involves many issues in addition to message passing, and really requires the definition of a standard for a complete distributed operating system for heterogeneous environments.

4. Other Standards Issues

As mentioned in the preceding subsection, ultimately it is desirable to define a standard for a distributed operating system. This is a more difficult undertaking than defining a standard for message-passing, and as mentioned at the workshop, involves important issues such as standards for parallel I/O. Other areas mentioned in which the development of standards would be beneficial include the definition of performance tracing routines and trace file formats, and standard tools for debugging, assessing performance and application behavior, etc.

It must also be decided whether the mapping of processes to physical processors is an issue that should be addressed in defining a message-passing standard. In many cases this reduces to assigning spatial subdomains to physical processors, and packages such as PARMACS provide quite sophisticated support for this task. The mapping issue is likely to be less important on "flat" machines for which the time to send a message between any two processors is only weakly dependent on their separation in the communication network. On non-flat machines, particularly when channel-addressed communication is used, the mapping of processes to processors has a significant impact on performance.

5. Summary

The general consensus emerging from the workshop was that now is a good time to begin the process of defining a standard for message-passing in distributed memory computing environments. To this end a Working Group of about 30 interested and public-spirited persons was formed, with Jack Dongarra serving as Chair and David Walker as Executive Director. The importance of involving European colleagues in defining the standard was stressed, and a number of Europeans are members of the Working Group. The main objective of the Working Group is to take the broad outline of a message-passing standard discussed in Sec. 3 and fashion it into a complete, well-defined, and practical standard. Rather than taking one of the existing message-passing systems and anointing it as the standard, the intent is to settle on the functional and semantic requirements (drawing where appropriate on existing systems for guidance), and then to define the detailed syntax of the standard. It is expected that the Working Group will meet about once every 4 to 6 months, and that it will take about 12 months to put forward a draft standard.

Appendix A. Workshop Program

The First CRPC Workshop on "Standards for Message Passing in a Distributed Memory Environment"

April 29–30, 1992

Hilton Conference Center
Williamsburg, Virginia, USA

Wednesday, April 29

First Session, 2:00pm to 3:15pm

- *"Message Passing Systems: Portability, Capability, Performance, Standards,"* Anthony Skjellum, Lawrence Livermore National Laboratory (30 min)
- *"European Initiatives Towards a Message Passing Standard,"* Rolf Hempel, GMD (30 min)
- Open Discussion (15 min)

Break, 3:15pm to 3:30pm

Second Session, 3:30pm to 5:30pm

- *"PICL: Description, Experiences, and Implications for Message-Passing Interface Standards,"* Patrick Worley, Oak Ridge National Laboratory (25 min)
- *"The Express Parallel Programming Environment,"* Jon Flower, Parasoft Corporation (25 min)
- *" Standards for Building Message Passing Systems Capable of Supporting Higher-Level Parallel Languages,"* Robert Bjornson, Scientific Computing Associates (25 min)
- *"Heterogeneous Distributed Computing with PVM,"* Adam Beguelin, University of Tennessee and Oak Ridge National Laboratory (25 min)
- Open Discussion (20 min)

Reception, 5:30pm to 7:30pm

Banquet, 7:30pm to 9:30pm

Thursday, April 30

Third Session, 8:30am to 10:30am

- *"Enhancements to NX/2 Message Passing for Portable Communications Libraries,"* Paul Pierce, Intel Corporation, Supercomputer Systems Division (25 min)
- *"Message Passing on the Vulcan Massively Parallel Computer,"* Vasanth Bala, IBM T. J. Watson Research Center (25 min)
- *"The Reactive Kernel and Cosmic Environment: Native and Emulated Systems for Medium-Grain Multicomputers and Workstation Networks,"* Anthony Skjellum, Lawrence Livermore National Laboratory (25 min)
- *"The CMMD Message Passing Library for the CM-5,"* Lew Tucker and Lennart Johnsson, Thinking Machines Corporation and Harvard University (25 min)
- Open Discussion (10 min)

Break, 10:30am to 10:40am

Fourth Session, 10:40am to 12:40pm

- *"Message-passing on CRAY Computer Systems,"* Peter Rigsbee, Cray Research, Inc. (25 min)
- *"The Computing Surface Network,"* Eric Barton, Meiko (25 min)
- *"Shared Objects and their Role in Standardization,"* Jonathan Nash, Leeds University (25 min)
- *"Low Latency Loosely Synchronous Communication Primitives,"* Matt Rosing, ICASE (25 min)
- *"Portable Programs for Parallel Processors: the P4 System,"* Ewing Lusk, Argonne National Laboratory (10 min)
- Open Discussion (10 min)

Lunch 12:40pm to 2:00pm

Fifth Session, 2:00pm to 3:50pm

- *"PARMACS: the ANL/GMD Portability Macros for Message Passing,"* Rolf Hempel, GMD (25min)
- *"MetaMP: A Higher Level Abstraction for Message Passing,"* Steve Otto, Oregon Graduate Institute (25 min)
- *"A Set of High Level Collective Communication Routines for Multicomputers,"* Robert van de Geijn, University of Texas at Austin (25 min)
- *"PVM++: An Object-Oriented Interface for Heterogeneous Computing,"* Roldan Pozo, University of Tennessee (25 min)
- Open Discussion (10 min)

Break, 3:50pm to 4:00pm
Sixth Session, 4:00pm to 5:00pm

- Panel Discussion (55 min)
  - Ken Kennedy, Rice University, moderator
  - Al Geist, Oak Ridge National Laboratory
  - Michael Heath, University of Illinois
  - Rolf Hempel, GMD
  - Anthony Skjellum, Lawrence Livermore National Laboratory
- Wrap-Up, David Walker and Jack Dongarra (5 min)

Workshop Ends, 5:00pm

Appendix B. List of Attendees

Given below is a list of the attendees at the First CRPC Workshop on "Standards for Message Passing in a Distributed Memory Environment," held April 29–30, 1992, at the Williamsburg Hilton, Virginia. A reasonable effort has been made to ensure that the information given here is correct, however, there are no doubt errors. It is hoped that these do not cause too much inconvenience.

Giovanni Aloisio
Dipt. di Elettrotecnica ed Elettronica
Universita di Bari
Via Re David 200
70125 Bari, ITALY
+39 80-241311 (phone)
+39 80-242410 (fax)
`gax%astrba.ba.cnr.it@icineca.cineca.it`

Ian G. Angus
Boeing Computer Services
M/S 7L-22
P. O. Box 24346
Seattle, WA 98124-0346
206 957-5853 (phone)
`angus@atc.boeing.com`

Marco Annaratone
Digital Equipment Corporation
146 Main Street MLO1-5/U46
Maynard, MA 01754
`marco_a@crl.dec.com`

Vasanth Bala
IBM T. J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598
914 945-1004 (phone)
914 945-2141 (fax)
`vas@watson.ibm.com`

Eric Barton
Meiko Limited
650 Aztec West
Bristol BS12 4SD
UNITED KINGDOM
+44 454-616171 (phone)
`eric@meiko.co.uk`

Adam Beguelin
Carnegie Mellon University
School of Computer Science
5000 Forbes Avenue
Pittsburgh, PA 15213-3890
412 268-5295 (phone)
`adamb@cs.cmu.edu`

Siegfried Benker
Institute for Statistics and Computer Science
University of Vienna
A-1210 Vienna
AUSTRIA
`sigi@par.univie.ac.at`

Roger Berry
NCUBE Corporation
4313 Prince Road
Rockville, MD 20853
`rogerb@ncube.com`

Scott Berryman
Yale University
Computer Science Department
51 Prospect Street
New Haven, CT 06520
203 432-1221 (phone)
`berryman@cs.yale.edu`

Robert Bjornson
Department of Computer Science
Box 2158 Yale Station
New Haven, CT 06520
203 432-1219 (phone)
`bjornson@cs.yale.edu`

Peter Brezany
Institute for Statistics and Computer Science
University of Vienna
A-1210 Vienna
AUSTRIA
`brezany@par.univie.ac.at`

Siddhartha Chatterjee
RIACS
Mail Stop T045-1
NASA Ames Research Center
Moffett Field, CA 94035-1000
415 604-4316 (phone)
415 604-3957 (fax)
`sc@riacs.edu`

Kuo-Ning Chiang
MacNeil-Schwendler Corporation
815 Colorado Blvd
Los Angeles, CA 90041
213 258-9111 (phone)
`k_chiang@macsch.com`

Jaeyoung Choi
Oak Ridge National Laboratory
Bldg. 6012 / MS-6367
P. O. Box 2008
Oak Ridge, TN 37831-6367
615 574-8696 (phone)
615 574-0680 (fax)
`choi@msr.epm.ornl.gov`

Mike Colajanni
Dip. di Ingegneria Elettronica
Universita' di Roma "Tor Vergata"
Via della Ricerca Scientifica
00133 – Roma
ITALY
+39-6-72594478 (phone)
+39-6-2020519 (fax)
`colajanni@tovvx1.ccd.utovrm.it`

Jack Dongarra
University of Tennessee
107 Ayres Hall
Department of Computer Science
Knoxville, TN 37996-1301
615 974-8295 (phone)
615 974-8296 (fax)
`dongarra@cs.utk.edu`

Tom Eidson
Theoretical Flow Physics Branch, M/S 156
NASA Langley Research Center
Hampton, VA 23665
804 864-2180 (phone)
804 865-6766 (fax)
`teidson@icase.edu`

Victor Eijkhout
University of Tennessee
107 Ayres Hall
Department of Computer Science
Knoxville, TN 37996-1301
`eijkhout@cs.utk.edu`

Rob Falgout
Lawrence Livermore National Lab
L-419
P. O. Box 808
Livermore, CA 94551
510 422-4377 (phone)
510 422-8920 (fax)
`rfalgout@llnl.gov`

Jim Feeney
IBM Endicott
R. D. 3, Box 224
Endicott, NY 13760
`feeneyj@gdlvm6.vnet.ibm.com`

Edward Felten
Department of Computer Science
University of Washington
Seattle, WA 98195
206 685-2675 (phone)
`felten@cs.washington.edu`

Vince Fernando
NAG Limited
Wilkinson House
Jordan Hill Road
Oxford, OX2 8DR
UNITED KINGDOM
+44 865-511245 (phone)
`fernando@cs.berkeley.edu`

Jon Flower
Parasoft Corporation
2500 E. Foothill Blvd.
Suite 205
Pasadena, CA91107
`jwf@elephant.parasoft.com`

Al Geist
Oak Ridge National Lab
Bldg. 6012 / MS-6367
P. O. Box 2008
Oak Ridge, TN 37831-63673
615 574-3153 (phone)
615 574-0680 (fax)
geist@msr.epm.ornl.gov

Mike Gerndt
Zentralinstitut fuer Angewandte Mathematik
Forschungszentrum Juelich GmbH
Postfach 1913
D-5170 Juelich
GERMANY
+49 2461-616569 (phone)
+49 2461-616656 (fax)
m.gerndt@kfa-juelich.de

Ian Glendinning
University of Southampton
Dept. of Electronics and Comp. Sci.
Southampton, SO9 5NH
UNITED KINGDOM
+44 703-593368 (phone)
+44 703-593045 (fax)
igl@ecs.soton.ac.uk

Adam Greenberg
Thinking Machines Corporation
245 First Street
Cambridge, MA 02142-1214
617 234-2006 (phone)
moose@think.com

Sanjay Gupta
ICASE
Mail Stop 132C
NASA Langley Research Center
Hampton, VA 23665-5225
gupta@icase.edu

Fred Gustavson
IBM T. J. Watson Research Center
Room 33-260
P. O. Box 218
Yorktown Heights, NY 10598
914 945-1980 (phone)
gustav@watson.ibm.com

Leslie Hart
R/E/FS5
325 Broadway
Boulder, CO 80303
hart@fsl.noaa.gov

Michael Heath
University of Illinois
NCSA, 4157 Beckman Institute
405 North Mathews Avenue
Urbana, IL 61801-2300
217 333-6268 (phone)
217 244-2909 (fax)
heath@ncsa.uiuc.edu

Rolf Hempel
GMD
Schloss Birlinghoven
Postfach 13 16
D-W-5205 Sankt Augustin 1
GERMANY
`gmap10@gmdzi.gmd.de`

Tom Henderson
R/E/FS5
325 Broadway
Boulder, CO 80303
303 497-7252 (phone)
`hender@fsl.noaa.gov`

S. Lennart Johnsson
Thinking Machines Corporation
245 First Street
Cambridge, MA 02142-1214
617 234-2100 (phone)
`johnsson@think.com`

Charles Jung
IBM Kingston
67LB/MS 614
Neighborhood Road
Kingston, NY 12401
914 385-1226 (phone)
`jung@kgnvma.vnet.ibm.com`

Ken Kennedy
Rice University
Department of Computer Science
P. O. Box 1892
Houston, TX 77251
713 285-5188 (phone)
`ken@rice.edu`

Charles Koelbel
Rice University
CITI/CRPC
P. O. Box 1892
Houston, TX 77251
713 285-5304 (phone)
713-285-5136 (fax)
`chk@cs.rice.edu`

Edward Kushner
Intel Corporation
15201 NW Greenbrier Parkway
Beaverton, OR 97006
503 629-7658 (phone)
`kushner@ssd.intel.com`

William Gropp
Argonne National Laboratory
Mathematics and Computer Science
9700 South Cass Avenue, MCS 221
Argonne, IL 60439-4844
`gropp@mcs.anl.gov`

John Lewis
Boeing Computer Services
Mail Stop 7L-21
P. O. Box 24346
Seattle, WA 98124-0346
206 865-3510 (phone)
`jglewis@atc.boeing.com`

Rusty Lusk
Argonne National Laboratory
Mathematics and Computer Science
9700 South Cass Avenue, MCS 221
Argonne, IL 60439-4844
`lusk@mcs.anl.gov`

Oliver McBryan
University of Colorado at Boulder
Department of Computer Science
Campus Box 425
Boulder, CO 80309-0425
303 665-0544 (phone)
`mcbryan@cs.colorado.edu`

Piyush Mehrotra
ICASE
Mail Stop 132C
NASA Langley Research Center
Hampton, VA 23665
804 864-2188 (phone)
`pm@icase.edu`

Paul Messina
California Institute of Technology
Mail Stop 158-79
1201 E. California Boulevard
Pasadena, CA 91125
818 356-3907 (phone)
818 584-5917 (fax)
`messina@zephyr.caltech.edu`

Jonathan Nash
Leeds University
School of Computer Studies
Leeds LS2 9JT
UNITED KINGDOM
+44 532-335473 (phone)
`nash@scs.leeds.ac.uk`

Mike Norman
Edinburgh Parallel Computing Centre
James Clerk Maxwell Building
The King's Buildings
Mayfield Road
Edinburgh, EH9 3JZ
UNITED KINGDOM
`mgn@dcs.ed.ac.uk`

Steve Otto
Oregon Graduate Institute
Department of Computer Sci. & Eng.
19600 NW von Neumann Drive
Beaverton OR 97006-1999
503 690-1486 (phone)
503 690-1029 (fax)
`otto@cse.ogi.edu`

Andrea Overman
NASA Langley Research Center
MS 125
Hampton, VA 23665
804 864-5790 (phone)
804 864-7635 (fax)
`overman@alosun.larc.nasa.gov`

David Payne
Intel Corporation
Supercomputer Systems Division
15201 NW Greenbrier Parkway
Beaverton, OR 97006
818 356-7573 (phone)
`payne@ccsf.caltech.edu`

Paul Pierce
Intel Corporation
Supercomputer Systems Division
15201 NW Greenbrier Parkway
Beaverton, OR 97006
`prp@ssd.intel.com`

Roldan Pozo
University of Tennessee
107 Ayres Hall
Department of Computer Science
Knoxville, TN 37996-1301
`pozo@cs.utk.edu`

Padma Raghavan
University of Illinois
NCSA, 4151 Beckman Institute
405 North Matthews Avenue
Urbana, IL 61801
217 244-3282 (phone)
`padma@ncsa.uiuc.edu`

Sanjay Ranka
Syracuse University
Northeast Parallel Architectures Center
111 College Place
Syracuse, NY 13244-4100
`ranka@top.cis.syr.edu`

Peter Rigsbee
Cray Research Incorporated
655 Lone Oak Drive
Eagan MN 55121
612 452-6650 (phone)
`par@cray.com`

Matt Rosing
ICASE
Mail Stop 132C
NASA Langley Research Center
Hampton, VA 23665-5225
`rosing@icase.edu`

Joel Saltz
ICASE
Mail Stop 132C
NASA Langley Research Center
Hampton, VA 23665-5225
804 864-2210 (phone)
804 864-6134 (fax)
`jhs@icase.edu`

Anthony Skjellum
Lawrence Livermore National Lab
L-316, P. O. Box 808
Livermore, CA 94550
510 422-1161 (phone)
510 423-2993 (fax)
`tony@helios.llnl.gov`

Steven G. Smith
Lawrence Livermore National Lab
L-419, P. O. Box 808
Livermore, CA 94550
510 293-8958 (phone)
`smith84@llnl.gov`

Charles H. Still
Lawrence Livermore National Lab
L-416, P. O. Box 808
Livermore, CA 94550
510 294-4171 (phone)
510 294-6933 (fax)
`still@llnl.gov`

Alan Sussman
ICASE
Mail Stop 132C
NASA Langley Research Center
Hampton, VA 23665-5225
`als@icase.edu`

Anne Trefethen
Engineering & Theory Center
Cornell University
Ithaca, NY 14853
607 254-4462 (phone)
`aet@cs.cornell.edu`

Lew Tucker
Thinking Machines Corporation
245 First Street
Cambridge, MA 02142-1214
617 234-2040 (phone)
`tucker@think.com`

Robert van de Geijn
University of Texas
Department of Computer Sciences
TAI 2.124
Austin, TX 78712
512 471-9720 (phone)
`rvdg@cs.utexas.edu`

David W. Walker
Oak Ridge National Laboratory
Bldg. 6012 / MS-6367
P. O. Box 2008
Oak Ridge, TN 37831-6367
615 574-7401 (phone)
615 574-0680 (fax)
`walker@msr.epm.ornl.gov`

Tammy Welcome
Lawrence Livermore National Lab
Massively Parallel Computing Initiative
L-416, P. O. Box 808
Livermore, CA 94550
510 422-4994 (phone)
`tsw@llnl.gov`

Jim West
IBM Corporation
MC 5600
3700 Bay Area Blvd.
Houston, TX 77058
713 282-8722 (phone)
`west@houvmscc.vnet.ibm.com`

Patrick Worley
Oak Ridge National Laboratory
Bldg. 6012 / MS-6367
P. O. Box 2008
Oak Ridge, TN 37831-6367
615 574-3128 (phone)
615 574-0680 (fax)
`worley@msr.epm.ornl.gov`

Mohammad Zubair
NASA Langley Research Center
Mail Stop 132C
Hampton, VA 23665
`zubair@fiddler.larc.nasa.gov`

ORNL/TM-12147

INTERNAL DISTRIBUTION

1. B. R. Appleton
2. J. Choi
3–4. T. S. Darland
5. E. F. D'Azevedo
6. J. J. Dongarra
7. G. A. Geist
8. L. J. Gray
9. M. R. Leuze
10. E. G. Ng
11. C. E. Oliver
12. B. W. Peyton
13–17. S. A. Raby
18. C. H. Romine
19. T. H. Rowan
20–24. R. F. Sincovec
25–29. D. W. Walker
30–34. R. C. Ward
35. P. H. Worley
36. Central Research Library
37. ORNL Patent Office
38. K-25 Applied Technology Library
39. Y-12 Technical Library
40. Laboratory Records - RC
41–42. Laboratory Records Department

EXTERNAL DISTRIBUTION

43. Giovanni Aloisio, Dipt. di Elettrotecnica ed Elettronica, Universita di Bari, Via Re David 200, 70125 Bari, ITALY

44. Ian G. Angus, Boeing Computer Services, M/S 7L-22, P. O. Box 24346, Seattle, WA 98124-0346

45. Marco Annaratone, Digital Equipment Corporation, 146 Main Street MLO1-5/U46, Maynard, MA 01754

46. Donald M. Austin, 6196 EECS Bldg., University of Minnesota, 200 Union Street, S.E., Minneapolis, MN 55455

47. Vasanth Bala, IBM T. J. Watson Research Center, P. O. Box 218, Yorktown Heights, NY 10598

48. Edward H. Barsis, Computer Science and Mathematics, P. O. Box 5800, Sandia National Laboratories, Albuquerque, NM 87185

49. Eric Barton, Meiko Limited, 650 Aztec West, Bristol BS12 4SD, UNITED KINGDOM

50. Adam Beguelin, Carnegie Mellon University, School of Computer Science, 5000 Forbes Avenue, Pittsburgh, PA 15213-3890

51. Siegfried Benker, Institute for Statistics and Computer Science, University of Vienna, A-1210 Vienna, AUSTRIA

52. Roger Berry, NCUBE Corporation, 4313 Prince Road, Rockville, MD 20853

53. Scott Berryman, Yale University, Computer Science Department, 51 Prospect Street, New Haven, CT 06520

54. Robert Bjornson, Department of Computer Science, Box 2158 Yale Station, New Haven, CT 06520

55. Peter Brezany, Institute for Statistics and Computer Science, University of Vienna, A-1210 Vienna, AUSTRIA

56. Roger W. Brockett, Wang Professor of EE & CS, Division of Applied Sciences, Harvard University, Cambridge, MA 02138

57. John Cavallini, Scientific Computing Staff, Applied Mathematical Sciences, Office of Energy Research, U.S. Department of Energy, Washington, DC 20585

58. Siddhartha Chatterjee, RIACS, Mail Stop T045-1, NASA Ames Research Center, Moffett Field, CA 94035-1000

59. Kuo-Ning Chiang, MacNeil-Schwendler Corporation, 815 Colorado Blvd, Los Angeles, CA 90041

60. Michele Colajanni, Dip. di Ingegneria Elettronica, Universita' di Roma "Tor Vergata", Via della Ricerca Scientifica, 00133 - Roma, ITALY

61. Jack Dongarra, University of Tennessee, 107 Ayres Hall, Department of Computer Science, Knoxville, TN 37996-1301

62. John J. Dorning, Department of Nuclear Engineering Physics, Thornton Hall, McCormick Road, University of Virginia, Charlottesville, VA 22901

63. Tom Eidson, Theoretical Flow Physics Branch, M/S 156, NASA Langley Research Center, Hampton, VA 23665

64. Victor Eijkhout, University of Tennessee, 107 Ayres Hall, Department of Computer Science, Knoxville, TN 37996-1301

65. Rob Falgout, Lawrence Livermore National Lab, L-419, P. O. Box 808, Livermore, CA 94551

66. Jim Feeney, IBM Endicott, R. D. 3, Box 224, Endicott, NY 13760

67. Edward Felten, Department of Computer Science, University of Washington, Seattle, WA 98195

68. Vince Fernando, NAG Limited, Wilkinson House, Jordan Hill Road, Oxford, OX2 8DR, UNITED KINGDOM

69. Jon Flower, Parasoft Corporation, 2500 E. Foothill Blvd., Suite 205, Pasadena, CA91107

70. Geoffrey C. Fox, Northeast Parallel Architectures Center, 111 College Place, Syracuse University, Syracuse, NY 13244-4100

71. J. Alan George, Vice President, Academic and Provost, Needles Hall, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1

72. Mike Gerndt, Zentralinstitut fuer Angewandte Mathematik, Forschungszentrum Juelich GmbH, Postfach 1913, D-5170 Juelich, GERMANY

73. Ian Glendinning, University of Southampton, Dept. of Electronics and Comp. Sci., Southampton, SO9 5NH, UNITED KINGDOM

74. Gene H. Golub, Department of Computer Science, Stanford University, Stanford, CA 94305

75. Adam Greenberg, Thinking Machines Corporation, 245 First Street, Cambridge, MA 02142-1214

76. Sanjay Gupta, ICASE, Mail Stop 132C, NASA Langley Research Center, Hampton, VA 23665-5225

77. Fred Gustavson, IBM T. J. Watson Research Center, Room 33-260, P. O. Box 218, Yorktown Heights, NY 10598

78. John Gustafson, 236 Wilhelm, Ames Laboratory, Iowa State University, Ames, IA 50011

79. Leslie Hart, R/E/FS5, 325 Broadway, Boulder, CO 80303

80. Michael Heath, University of Illinois, NCSA, 4157 Beckman Institute, 405 North Mathews Avenue, Urbana, IL 61801-2300

81. Rolf Hempel, GMD, Schloss Birlinghoven, Postfach 13 16, D-W-5205 Sankt Augustin 1, GERMANY

82. Tom Henderson, R/E/FS5, 325 Broadway, Boulder, CO 80303

83. S. Lennart Johnsson, Thinking Machines Corporation, 245 First Street, Cambridge, MA 02142-1214

84. Charles Jung, IBM Kingston, 67LB/MS 614, Neighborhood Road, Kingston, NY 12401

85. Malvyn H. Kalos, Cornell Theory Center, Engineering and Theory Center Bldg., Cornell University, Ithaca, NY 14853-3901

86. Hans Kaper, Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Bldg. 221, Argonne, IL 60439

87. Ken Kennedy, Rice University, Department of Computer Science, P. O. Box 1892, Houston, TX 77251

88. Charles Koelbel, Rice University, CITI/CRPC, P. O. Box 1892, Houston, TX 77251

89. Edward Kushner, Intel Corporation, 15201 NW Greenbrier Parkway, Beaverton, OR 97006

90. William Gropp, Argonne National Laboratory, Mathematics and Computer Science, 9700 South Cass Avenue, MCS 221, Argonne, IL 60439-4844

91. James E. Leiss, Rt. 2, Box 142C, Broadway, VA 22815

92. John Lewis, Boeing Computer Services , Mail Stop 7L-21, P. O. Box 24346, Seattle, WA 98124-0346

93. Rusty Lusk, Argonne National Laboratory, Mathematics and Computer Science, 9700 South Cass Avenue, MCS 221, Argonne, IL 60439-4844

94. Oliver McBryan, University of Colorado at Boulder, Department of Computer Science, Campus Box 425, Boulder, CO 80309-0425

95. James McGraw, Lawrence Livermore National Laboratory, L-306, P.O. Box 808, Livermore, CA 94550

96. Piyush Mehrotra, ICASE, Mail Stop 132C, NASA Langley Research Center, Hampton, VA 23665

97. Paul Messina, California Institute of Technology, Mail Stop 158-79, 1201 E. California Boulevard, Pasadena, CA 91125

98. Neville Moray, Department of Mechanical and Industrial Engineering, University of Illinois, 1206 West Green Street, Urbana, IL 61801

99. Jonathan Nash, Leeds University, School of Computer Studies, Leeds LS2 9JT, UNITED KINGDOM

100. Mike Norman, Edinburgh Parallel Computing Centre, James Clerk Maxwell Building, The King's Buildings , Mayfield Road, Edinburgh, EH9 3JZ, UNITED KINGDOM

101. James M. Ortega, Department of Applied Mathematics, Thornton Hall, University of Virginia, Charlottesville, VA 22901

102. Steve Otto, Oregon Graduate Institute, Department of Computer Sci. & Eng., 19600 NW von Neumann Drive, Beaverton, OR 97006-1999

103. Andrea Overman, NASA Langley Research Center, MS 125, Hampton, VA 23665

104. David Payne, Intel Corporation, Supercomputer Systems Division, 15201 NW Greenbrier Parkway, Beaverton, OR 97006

105. Paul Pierce, Intel Corporation, Supercomputer Systems Division, 15201 NW Greenbrier Parkway, Beaverton, OR 97006

106. Robert J. Plemmons, Departments of Mathematics and Computer Science, Box 7311, Wake Forest University Winston-Salem, NC 27109

107. Roldan Pozo, University of Tennessee, 107 Ayres Hall, Department of Computer Science, Knoxville, TN 37996-1301

108. Padma Raghavan, University of Illinois, NCSA, 4151 Beckman Institute, 405 North Matthews Avenue, Urbana, IL 61801

109. Sanjay Ranka, Syracuse University, Northeast Parallel Architectures Center, 111 College Place, Syracuse, NY 13244-4100

110. Werner C. Rheinboldt, Department of Mathematics and Statistics, University of Pittsburgh, Pittsburgh, PA 15260

111. Peter Rigsbee, Cray Research Incorporated, 655 Lone Oak Drive, Eagan, MN 55121

112. Matt Rosing, ICASE, Mail Stop 132C, NASA Langley Research Center, Hampton, VA 23665-5225

113. Joel Saltz, ICASE, Mail Stop 132C, NASA Langley Research Center, Hampton, VA 23665-5225

114. Ahmed H. Sameh, Center for Supercomputing R&D, 1384 W. Springfield Avenue, University of Illinois, Urbana, IL 61801

115. David S. Scott, Intel Scientific Computers, 15201 N.W. Greenbrier Parkway, Beaverton, OR 97006

116. Anthony Skjellum, Lawrence Livermore National Lab, L-316, P. O. Box 808, Livermore, CA 94550

117. Steven G. Smith, Lawrence Livermore National Lab, L-419, P. O. Box 808, Livermore, CA 94550

118. Charles H. Still, Lawrence Livermore National Lab, L-416, P. O. Box 808, Livermore, CA 94550

119. Alan Sussman, ICASE, Mail Stop 132C, NASA Langley Research Center, Hampton, VA 23665-5225

120. Paul N. Swartztrauber, National Center for Atmospheric Research, P.O. Box 3000, Boulder, CO 80307

121. Anne Trefethen, Engineering & Theory Center, Cornell University, Ithaca, NY 14853

122. Lew Tucker, Thinking Machines Corporation, 245 First Street, Cambridge, MA 02142-1214

123. Robert van de Geijn, University of Texas, Department of Computer Sciences , TAI 2.124, Austin, TX 78712

124. Tammy Welcome, Lawrence Livermore National Lab, Massively Parallel Computing Initiative, L-416, P. O. Box 808, Livermore, CA 94550

125. Jim West, IBM Corporation, MC 5600, 3700 Bay Area Blvd., Houston, TX 77058

126. Mary F. Wheeler, Rice University, Department of Mathematical Sciences, P.O. Box 1892, Houston, TX 77251

127. Andrew B. White, Computing Division, Los Alamos National Laboratory, P.O. Box 1663, MS-265, Los Alamos, NM 87545

128. Mohammad Zubair, NASA Langley Research Center, Mail Stop 132C, Hampton, VA 23665

129. Office of Assistant Manager for Energy Research and Development, U.S. Department of Energy, Oak Ridge Operations Office, P.O. Box 2001 Oak Ridge, TN 37831-8600

130–139. Office of Scientific & Technical Information, P.O. Box 62, Oak Ridge, TN 37831