Skip to main content

- Home
- Topics
- Reference
- Glossary
- Help
- Notebook

Virtual Workshop

Welcome guest
Log in (Globus)
Log in (other)
Try the quiz before you start
MPI Collective Communications
Introduction Goals Prerequisites
Characteristics Three Types of Routines Barrier Synchronization Data Movement • Broadcast • Gather and Scatter • Gather/Scatter Effect • Gatherv and Scatterv • Allgather • All to All Global Computing • Reduce • Scan • Operations and Example • Allreduce Mini-Exercise Nonblocking Routines • Nonblocking Example Performance Issues • Two Ways to Broadcast • Two Ways to Scatter Application Example • Scatter vs. Scatterv • Scatterv Syntax
Exercise Quiz
Short survey

MPI Collective Communications: Broadcast

In many instances, there is one process that needs to send (broadcast) some data (either a scalar or vector) to all the processes in a group. MPI provides the broadcast primitive MPI_Bcast to accomplish this task.The syntax of the MPI_Bcast call is:

C

```
int MPI_Bcast(void* buffer, int count, MPI_Datatype datatype,
     int root, MPI_Comm comm)
```

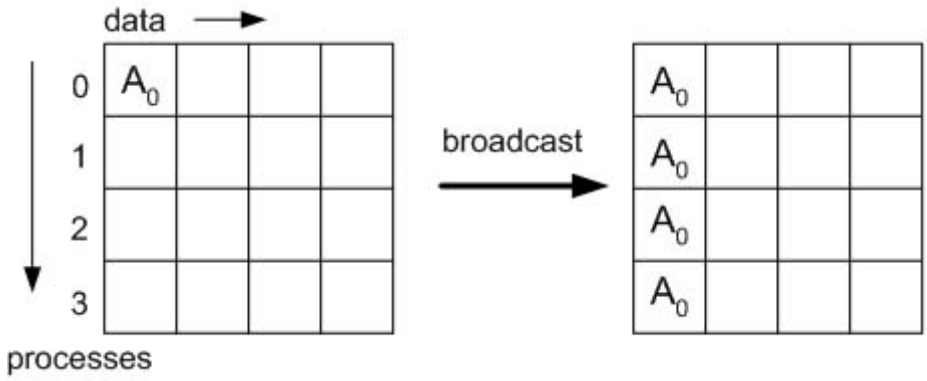
FORTRAN

```
MPI_BCAST(buffer, count, datatype, root, comm, ierr)
```

where:

buffer	is the starting address of a buffer,
count	is an integer indicating the number of data elements in the buffer,
datatype	is an MPI defined constant indicating the data type of the elements in the buffer,
root	is an integer indicating the rank of broadcast root process, and
comm	is the communicator.

The MPI_Bcast must be called by each process in the group, specifying the same comm and root. The message is sent from the root process to all processes in the group, including the root process. If we think in terms of a matrix whose rows and columns correspond to the data and processes, respectively, then the states of this matrix before and after the call can be illustrated as follows:



The “Hello, world” example used in earlier modules could have used broadcast instead of multiple sends and receives:

C Example:

```
#include <stdio.h>
#include <string.h>
#include "mpi.h"
int main(int argc, char **argv)
{
  char message[20];
  int  i, rank, size;
  MPI_Status status;
  int root = 0;

  MPI_Init(&argc, &argv);
  MPI_Comm_size(MPI_COMM_WORLD, &size);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);

  if (rank == root)
  {
    strcpy(message, "Hello, world");
  }
  MPI_Bcast(message, 13, MPI_CHAR, root, MPI_COMM_WORLD);
  printf( "Message from process %d : %s\n", rank, message);

  MPI_Finalize();
}
```

Fortran Example:

```
use MPI
character(12) message
integer rank,root
data root/0/
call MPI_INIT(ierr)
call MPI_COMM_RANK(MPI_COMM_WORLD, rank, ierr)
if (rank .eq. root) then
    message = "Hello, world"
endif
```

```
call MPI_BCAST(message, 12, MPI_CHARACTER, root, &
         MPI_COMM_WORLD, ierr)
print*, "node", rank, ":", message
call MPI_FINALIZE(ierr)
end
```

<= previous next =>

Add my notes
Mark (M) my place in this topic