MPI_Allreduce

Combines values from all processes and distribute the result back to all processes

Synopsis

```
#include "mpi.h"
int MPI_Allreduce ( sendbuf, recvbuf, count, datatype, op, comm )
void             *sendbuf;
void             *recvbuf;
int               count;
MPI_Datatype      datatype;
MPI_Op            op;
MPI_Comm          comm;
```

Input Parameters

sendbuf - starting address of send buffer (choice)

count - number of elements in send buffer (integer)

datatype - data type of elements of send buffer (handle)

op - operation (handle)

comm - communicator (handle)

Output Parameter

recvbuf - starting address of receive buffer (choice)

Notes for Fortran

All MPI routines in Fortran (except for `MPI_WTIME` and `MPI_WTICK`) have an additional argument `ierr` at the end of the argument list. `ierr` is an integer and has the same meaning as the return value of the routine in C. In Fortran, MPI routines are subroutines, and are invoked with the `call` statement.

All MPI objects (e.g., `MPI_Datatype`, `MPI_Comm`) are of type `INTEGER` in Fortran.

Notes on collective operations

The reduction functions (`MPI_Op`) do not return an error value. As a result, if the functions detect an error, all they can do is either call `MPI_Abort` or silently skip the problem. Thus, if you change the error handler from `MPI_ERRORS_ARE_FATAL` to something else, for example, `MPI_ERRORS_RETURN`, then no error may be indicated.

The reason for this is the performance problems in ensuring that all collective routines return the same error value.

Errors

All MPI routines (except `MPI_Wtime` and `MPI_Wtick`) return an error value; C routines as the value of the function and Fortran routines in the last argument. Before the value is returned, the current MPI error handler is called. By default, this error handler aborts the MPI job. The error handler may be changed with `MPI_Errhandler_set`; the predefined error handler `MPI_ERRORS_RETURN` may be used to cause error values to be returned. Note that MPI does *not* guarentee that an MPI program can continue past an error.

MPI_ERR_BUFFER - Invalid buffer pointer. Usually a null buffer where one is not valid.

MPI_ERR_COUNT - Invalid count argument. Count arguments must be non-negative; a count of zero is often valid.

MPI_ERR_TYPE - Invalid datatype argument. May be an uncommitted

MPI_Datatype (see `MPI_Type_commit`).

MPI_ERR_OP - Invalid operation. MPI operations (objects of type `MPI_Op`) must either be one of the predefined operations (e.g., `MPI_SUM`) or created with `MPI_Op_create`.

MPI_ERR_COMM - Invalid communicator. A common error is to use a null communicator in a call (not even allowed in `MPI_Comm_rank`).

Location:allreduce.c