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: Gatherv and Scatterv

MPI_Gatherv and MPI_Scatterv are the variable-message-size versions of MPI_Gather and MPI_Scatter. MPI_Gatherv extends the functionality of MPI_Gather to permit a varying count of data from each process, and to allow some flexibility in where the gathered data is placed on the root process. It does this by changing the `count` argument from a single integer to an integer array and providing a new argument `displs` (an array) . MPI_Scatterv extends MPI_Scatter in a similar manner. More information on the use of these routines will be presented in an Application Example later in this module.

C

```
int MPI_Gatherv(const void* sbuf, int scount, \
     MPI_Datatype stype, void* rbuf, const int rcounts[], \
     const int displs[], MPI_Datatype rtype, \
     int root, MPI_Comm comm)

int MPI_Scatterv(const void* sbuf, const int scounts[], \
     const int displs[], MPI_Datatype stype, void* rbuf, \
     int rcount, MPI_Datatype rtype, \
     int root, MPI_Comm comm)
```

FORTRAN

```
MPI_GATHERV(sbuf, scount, stype, rbuf, rcounts, displs, rtype,
     root, comm, ierr)
```

```
MPI_SCATTERV(sbuf, scounts, displs, stype, rbuf, rcount, rtype,
     root, comm, ierr)
```

The variables for **Gatherv** are:

sbuf	starting address of send buffer,
scount	number of elements in send buffer,
stype	data type of send buffer elements,
rbuf	starting address of receive buffer,
rcounts	array containing number of elements to be received from each process,
displs	array specifying the displacement relative to rbuf at which to place the incoming data from corresponding process,
rtype	data type of receive buffer,
root	rank of receiving process,
comm	group communicator.

Note: rbuf, rcounts, displs, rtype are significant for the root process only.

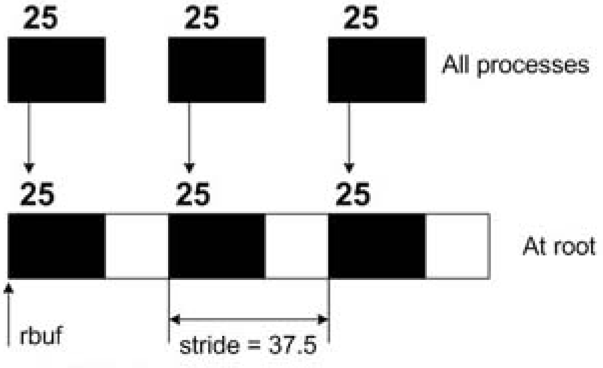
The variables for **Scatterv** are:

sbuf	address of send buffer,
scounts	integer array specifying the number of elements to send to each process,
displs	array specifying the displacement relative to sbuf from which to take the data going out to the corresponding process,
stype	data type of send buffer elements,
rbuf	address of receive buffer,
rcount	number of elements in receive buffer,
rtype	data type of receive buffer elements,
root	rank of sending process,
comm	group communicator

Note: sbuf, scounts, displs, stype are significant for the root process only.

For the purpose of illustrating the usage of MPI_Gatherv and MPI_Scatterv, we give two Fortran program fragments below:

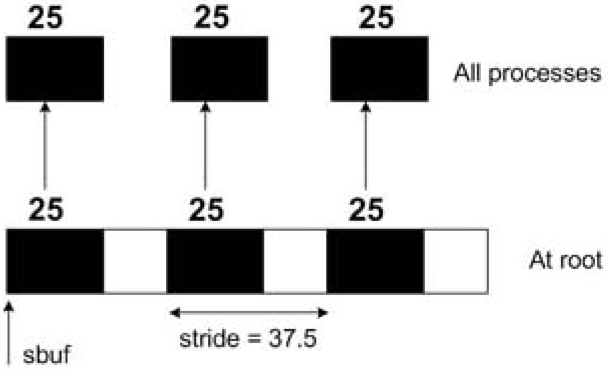
MPI_GATHERV Example



```
real a(25), rbuf(MAX)
integer displs(NX), rcounts(NX), nsize
...
do i= 1, nsize
   displs(i) = (i-1) * stride
   rcounts(i) = 25
enddo
call mpi_gatherv(a, 25, MPI_REAL, rbuf, rcounts, displs, &
      MPI_REAL, root, comm, ierr)
...
```

Notice that the effect of setting stride= 37.5 is to cause the separation between the chunks on the root to alternate between 12 and 13 integers.

MPI_SCATTERV Example



```
real a(25), sbuf(MAX)
integer displs(NX), scounts(NX), nsize
...
do i= 1, nsize
   displs(i) = (i-1) * stride
   scounts(i) = 25
enddo
call mpi_scatterv(sbuf, scounts, displs, MPI_REAL, a, 25, &
      MPI_REAL, root, comm, ierr)
...
```

Notice again that the effect of setting stride=37.5 is to cause the separation between the chunks on the root to alternate between 12 and 13 integers.

<= previous next =>

Add my notes

Mark (M) my place in this topic